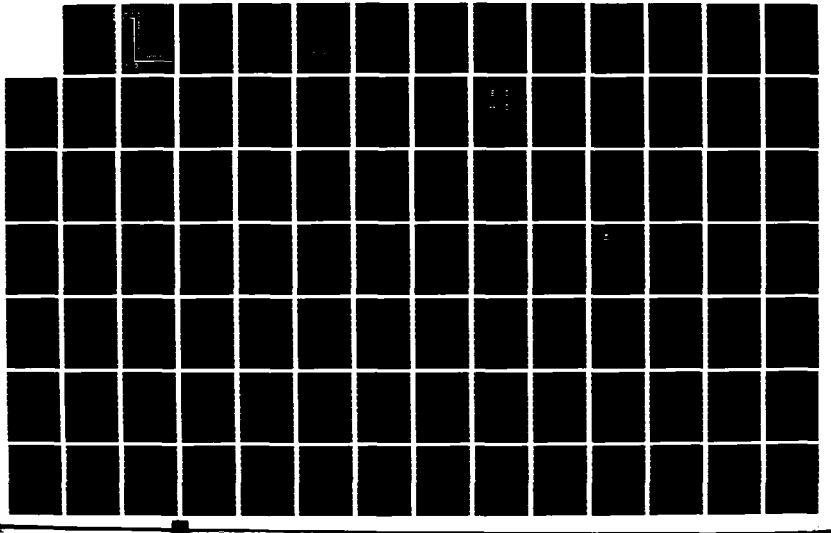
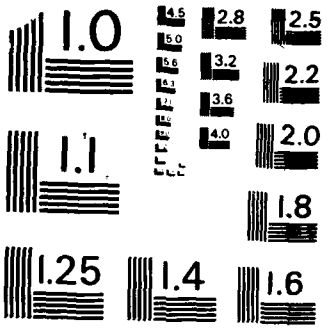
AD-A175 235 MISSION RELIABILITY MODEL USERS GUIDE(U) ANALYTIC SCIENCES CORP READING MA M H VEATCH ET AL. NOV 86 AFHRL-TR-86-35 F19628-82-C-0002 1/2

UNCLASSIFIED F/G 14/4 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS - 1963 - A

AFHRL-TR-86-35

(12)

AIR FORCE HUMAN RESOURCES LABORATORY

AD-A175 235

DTIC FILE COPY

MISSION RELIABILITY MODEL USERS GUIDE

Michael H. Veatch
Robert K. Gates
The Analytic Sciences Corporation
One Jacob Way
Reading, Massachusetts 01867

LOGISTICS AND HUMAN FACTORS DIVISION
Wright-Patterson Air Force Base, Ohio 45433-6503

November 1986
Final Report for Period March 1982 - March 1986

Approved for public release; distribution is unlimited.

DTIC
ELECTE
DEC 18 1986
S D
B

AIR FORCE SYSTEMS COMMAND
BROOKS AIR FORCE BASE, TEXAS 78235-5601

86 1 25

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S) AFHRL-TR-86-35
6a. NAME OF PERFORMING ORGANIZATION The Analytic Sciences Corporation 6b. OFFICE SYMBOL (*If applicable*)	7a. NAME OF MONITORING ORGANIZATION Logistics and Human Factors Division
6c. ADDRESS (*City, State, and ZIP Code*) One Jacob Way Reading, Massachusetts 01867	7b. ADDRESS (*City, State, and ZIP Code*) Air Force Human Resources Laboratory Wright-Patterson Air Force Base, Ohio 45433-6503
8a. NAME OF FUNDING / SPONSORING ORGANIZATION Air Force Human Resources Laboratory 8b. OFFICE SYMBOL (*If applicable*) HQ AFHRL	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER F33615-82-C-0002
8c. ADDRESS (*City, State, and ZIP Code*) Brooks Air Force Base, Texas 78235-5601	10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.
62205F	1710	00	26

11. TITLE (*Include Security Classification*)
Mission Reliability Model Users Guide

12. PERSONAL AUTHOR(S)
Veatch, Michael H.; Gates, Robert K.

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (*Year, Month, Day*)	15. PAGE COUNT
Final	FROM Mar 82 TO Mar 86	November 1986	142

16. SUPPLEMENTARY NOTATION

17. COSATI CODES		
FIELD	GROUP	SUB-GROUP
15	05	
12	01	

18. SUBJECT TERMS (*Continue on reverse if necessary and identify by block number*)
communication; fault-tolerant avionics; identification; logistics analysis; logistics models; mean time between critical failure (Continued)

19. ABSTRACT (*Continue on reverse if necessary and identify by block number*)

The Mission Reliability Model (MIREM) has been developed to evaluate the reliability and sustained operating capability of advanced electronic systems during the early stages of design. MIREM is applicable to integrated electronic systems that achieve fault tolerance through dynamic fault detection, fault isolation, and reconfiguration.

The newest version of MIREM also has the capability to examine the effects of false alarms, undetected failures, and innovative repair policies on system reliability. This report, which includes sample input and output files, thoroughly discusses the techniques used in MIREM, along with possible model applications.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	

22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE (*Include Area Code*)	22c. OFFICE SYMBOL
Nancy A. Perrigo, Chief, STINFO Office	(512) 536-3877	AFHRL/TSR

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
Unclassified

Item 18 (Concluded):

mean time between failure
mean time to repair
mission completion success probability
system reliability

SUMMARY

The Mission Reliability Model (MIREM) has been developed to evaluate the reliability and sustained operating capability of advanced electronic circuits during the early stages of development. MIREM is applicable to integrated systems that achieve fault tolerance through dynamic fault detection, fault isolation, and reconfiguration. The model can also be valuable in evaluating designs that make use of "hard wired" or "brute force" redundancy.

The most unique feature of MIREM is its ability to accurately reflect the impact of reconfigurable, competing functions on system reliability. The user defines the resources necessary to support a required function (e.g., Identify Friend/Foe, IFF), and the model will compute the probability of losing that functional capability over a certain operating time. A critical failure occurs when there are not a sufficient number of working resources to support a certain function. As an analytic model, MIREM determines a specific value for Mean Time Between Critical Failure, Mission Completion Success Probability, and Failure Resiliency.

This report documents the latest enhancements added to MIREM. The model can now calculate the effects of undetected failures and false alarms upon system reliability. Mean Time Between Maintenance Action and Mean Time to Repair are two output statistics which have been added. Graphical outputs, now included with MIREM, illustrate repair options for circuits where reliability gracefully degrades over a period of time.

This users guide describes all features of the model. Sample problems are provided, along with example computer runs, which illustrate the different ways in which MIREM can be used.

DTIC
ELECTE
DEC 1 8 1986
B

Accession For
NTIS GRA&I
DTIC TAB
Unannounced
Justification
By
Distribution
Availability
Dist Special
A-1

PREFACE

This report provides user documentation for MIREM, a program for evaluating the reliability of advanced fault-tolerant systems in a mission scenario. It is consistent with the software release dated 17 March 1986, and replaces earlier documentation dated 23 August 1985.

The author wishes to recognize Dr. Robert Foley of the Georgia Institute of Technology, who developed several of the algorithms used in MIREM, and to thank his associates at TASC, Mssrs. Peter G. Clark, Joseph Medina, and Jonathan H. Simonson, who were responsible for developing the MIREM software. This work is sponsored by the Air Force Human Resources Laboratory. The guidance and support of Lt Lee Dayton of this laboratory are greatly appreciated.

TABLE OF CONTENTS

		Page
1.	INTRODUCTION	1
	1.1 Overview	1
	1.2 Scope	3
	1.3 Evolution of the Model	4
	1.4 Organization of the Users Guide	5
2.	RELIABILITY MODEL	6
	2.1 Measures of Effectiveness	6
	2.2 Mission Scenarios	8
	2.3 MIREM Computational Approach	8
	2.3.1 Resource Failure Model	8
	2.3.2 System Failure Model	9
	2.3.3 Resource Requirements Model	12
	2.3.4 Repair Model	13
	2.3.5 Imperfect Switching Model	14
	2.4 Reliability Block Diagram Interpretation of MIREM	16
3.	INPUT DATA PREPARATION	17
	3.1 Overview of Data Requirements	18
	3.2 Preparing the Function List	22
	3.3 Preparing the Resource List	22
	3.4 Identifying Resource Chains	24
	3.5 Identifying Resource Pools	25
	3.6 Identifying Resource Groups	34
	3.7 Preparing the LRM/LRU List	37
	3.8 Using Reliability Block Diagrams	38
4.	PROGRAM OPERATING PROCEDURES	38
	4.1 Program Overview	38
	4.2 DATAIN Program Operation	41

TABLE OF CONTENTS (Continued)

		Page
4.2.1	Initiating DATAIN	41
4.2.2	DATAIN Keywords	41
4.2.3	File Maintenance	42
4.2.4	Architecture File Dialogue	45
4.2.5	Scenario File Dialogue	48
4.3	Direct Editing Option	50
4.3.1	Architecture File Format	50
4.3.2	Scenario File Format	54
4.4	MIREM Program Operation	56
4.5	MPLOT Program Operation	57
4.6	Limitations	57
4.7	Compatibility of Features	58
5.	SAMPLE STUDIES	60
5.1	Mission Effectiveness Analysis	60
5.2	Logistics/Sustainability Analysis	64
5.3	Repair Policy Analysis	68
5.4	Sensitivity Analysis	69
6.	ADVANCED APPLICATIONS	70
6.1	Resource Pool Approximations	70
6.1.1	Branches Differ	70
6.1.2	Pool Boundaries Differ	70
6.1.3	Singleton Type C Pools	71
6.2	Modeling Resource Scheduling with Utilization Rates	71
6.3	Resource Chain Approximations	72
6.3.1	General Series/Parallel Structures	73
6.3.2	Chain Boundaries Differ	73
APPENDIX A:	MIREM EQUATIONS	75
APPENDIX B:	SAMPLE SESSION LISTING	108
APPENDIX C:	DATA ENTRY FORMS	127
APPENDIX D:	GLOSSARY	130

LIST OF FIGURES

Figure		Page
1	Overview of Analysis Using MIREM	2
2	Fault Tolerance Concepts	3
3	A Two-Level Structure for System Architecture Representation	10
4	Examples of Series/Parallel Structures	11
5	Function Descriptions for an Example Architecture	19
6	Processing Descriptions for an Example Architecture	20
7	Block Diagram for an Example Architecture	21
8	Fault Recovery Techniques for an Example Architecture	21
9	Function List for the Example Architecture . . .	22
10	Resource List for the Example Architecture . . .	23
11	Resource Chains for the Example Architecture . .	25
12	Chain Data for the Example Architecture	26
13	Pool Data Entry Form for the Example Architecture	31
14	Pool and Chain Structure for the Example Architecture	35
15	Example Series-Parallel Structure	36
16	Pool and Group Data for Figure 15	37
17	LRM/LRU List for the Example Architecture . . .	38

LIST OF FIGURES (Continued)

Figure		Page
18	UHF RBD for the Example Architecture	39
19	MIREM Program Overview	40
20	DATAIN Screen Flowchart	42
21	DATAIN Control Keywords	43
22	Overview of DATAIN Dialogue	44
23	Architecture File Menu Screen	45
24	Pool List Screen	46
25	Pool Data Entry Screen	47
26	Functions in Chain Screen	48
27	Architecture File Format	50
28	Architecture File for the Example Architecture. .	51
29	Scenario File Format	55
30	Sample Scenario File	55
31	Architecture File Report for the Example Architecture	61
32	Scenario File Report	62
33	MCSP and Pool Budget Output	63
34	Phase-by-Phase MCSP Output	63
35	Testability Factors (BIT) Output	64
36	LRM/LRU Budget Output	64
37	MTBCF Output	65
38	Failure Rate as a Function of Operating Time . .	66

LIST OF FIGURES (Continued)

Figure		Page
39	Testability Factors (BIT) MTBCF Output	67
40	MTBFF Output	67
41	Repair Output Option	68

LIST OF TABLES

Table		Page
1	New Features	5
2	Processing Allocations for an Example Architecture	19
3	Criteria for Identifying Resource Chains	26
4	Resource Utilization Worksheet	28
5	Criteria for Identifying Resource Pools	29
6	Criteria for Assigning Pool Types	30
7	Scenario File Run Parameters	49
8	MIREM Size Limitations	58
9	Compatibility of Features	59
10	Sensitivity of MCSP to Configuration Changes . .	69
11	Fractional Utilizations for a Partially Contending Resource Pool	72

1. INTRODUCTION

1.1 Overview

The Mission Reliability Model (MIREM) is a fault-tolerant system reliability program developed to evaluate the mission reliability, availability, and sustained operating capability of advanced electronics systems early in their development phase. These systems contain integration, redundancy, and dynamic reconfigurability (self-repair) as part of their fault-tolerant design. Typical analyses that can be conducted using MIREM include:

1. Evaluation of mission reliability for alternative mission scenarios.

2. Determination of the additional operating time without repair that can be achieved in fault-tolerant systems in comparison with conventional systems.

3. Identification of the parts within a system that are contributing significantly to mission failures.

4. Identification of design improvements that offer a large payoff in mission reliability.

5. Determination of the increased availability that can be achieved using deferred maintenance policies.

These analysis capabilities are provided by a new mathematical model constructed to assess a broad class of fault-tolerant structures.

Program MIREM is written in FORTRAN 77 for operation in a large variety of computer installations. Interactive, full-screen input-output sessions facilitate convenient operation. Graphical output is provided using DI3000, for users with graphics terminals or plotters and DI3000 software. Figure 1 shows the analysis process using MIREM. System description data available during the development phase are used to prepare the fault-tolerant structure data used by MIREM. Baseline architecture and scenario files are then created using the data entry program. A number of run and output options are selected by the analyst and stored in scenario files, which are then executed by the computational program. Graphical

outputs can then be generated by running the plotting program. The analytic (as opposed to simulation) approach taken in the mathematical model results in very fast run times for most scenarios and enhances the program's utility for iterative "what if" investigations.

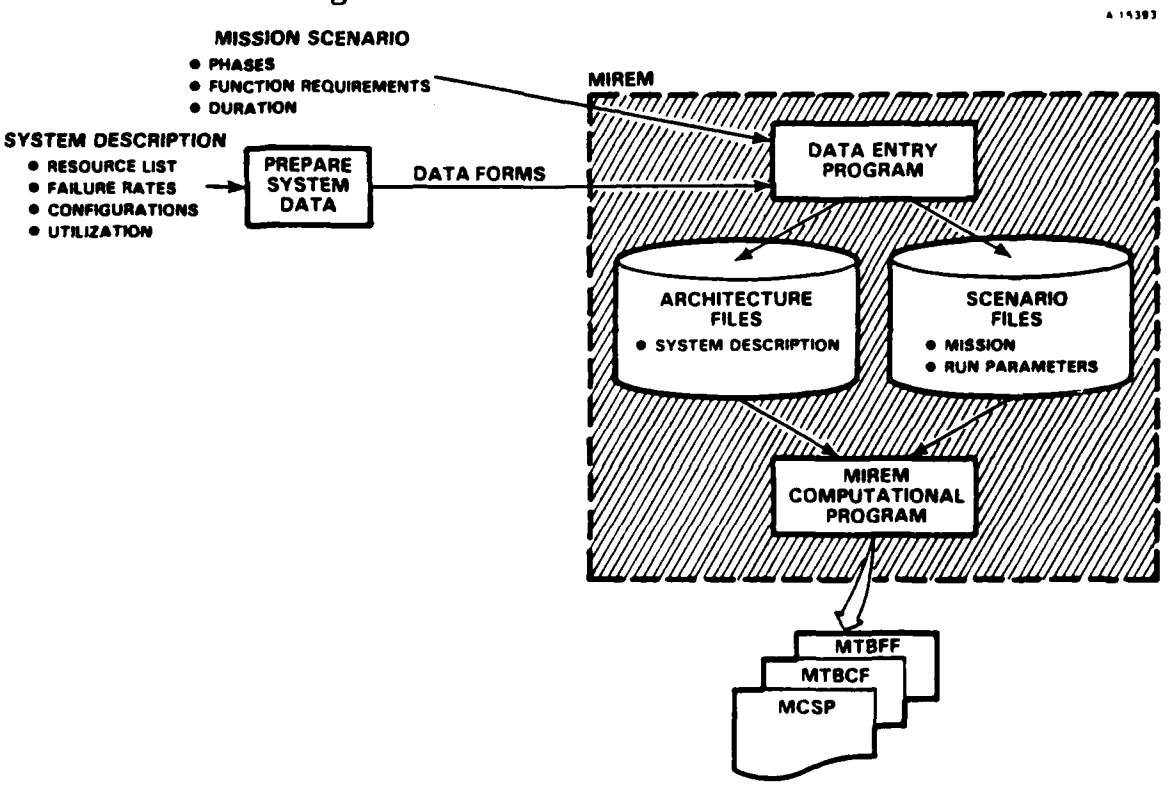


Figure 1. Overview of Analysis Using MIREM.

The MIREM methodology addresses many of the major reliability issues for advanced systems:

1. Redundancy at the Line Replaceable Unit (LRU), Line Replaceable Module (LRM), and component levels.

2. Dynamic reconfigurability and resource sharing that allow several system functions to use the same components for primary or backup operation.

3. Fault recovery software that allows processing to resume when computer faults occur.

4. Imperfect switching resulting from incomplete Built-In Test (BIT) coverage.

5. Modular packaging that influences redundancy and fault isolation levels.

6. Deferred repair policies that exploit fault tolerance to improve availability.

7. Meaningful measures of effectiveness for multifunction systems in a mission environment.

The methodology also applies to more conventional redundant systems. The model does not estimate component failure rates or BIT coverage; reliability estimates derived from MIL-HDBK-217D, parametric analysis for Very High Speed Integrated Circuits (VHSIC) parts, or by other techniques are accepted as model inputs. Software errors are also not considered in the model.

1.2 Scope

MIREM is applicable to advanced integrated electronic systems that achieve fault tolerance through dynamic fault detection, fault isolation, and reconfiguration (Figure 2). This dynamic process allows failed items to be replaced by backups or items which were originally assigned to other

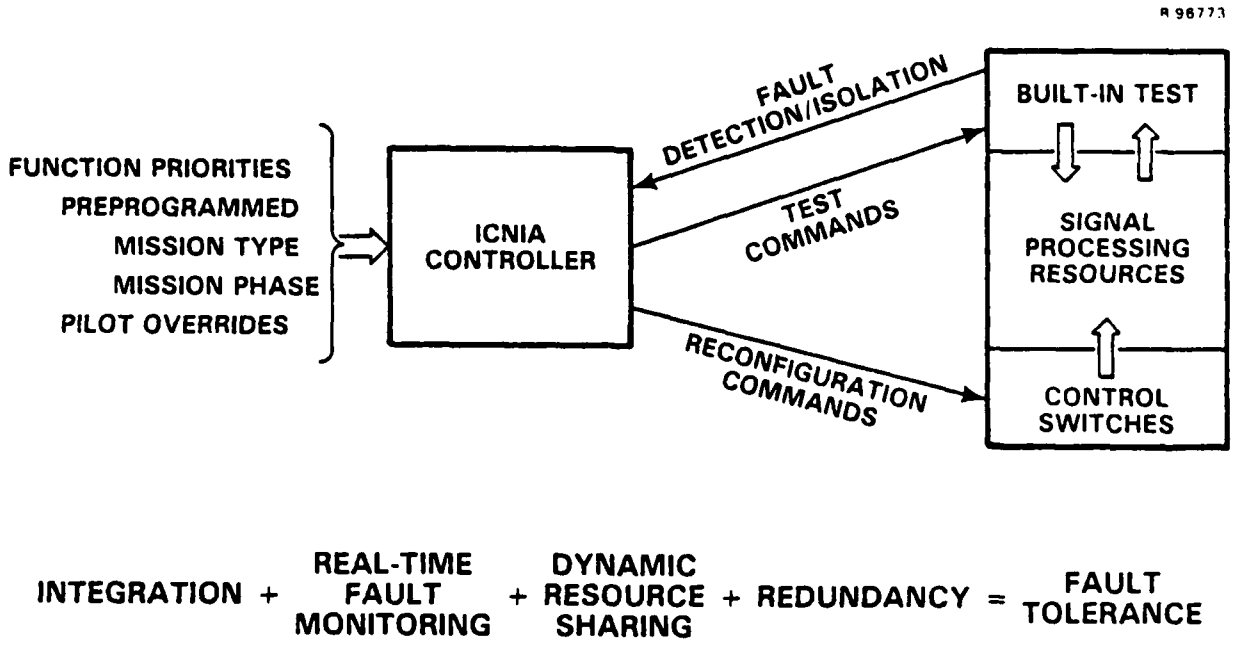


Figure 2. Fault Tolerance Concepts.

functions. Fault detection/isolation is performed by BIT equipment, which isolates faults to the lowest "failure unit," referred to as a resource. A system controller tracks function requirements and system health in terms of failed or good resources. When a failure occurs or requirements change, the controller will reconfigure the system in an attempt to meet the function requirements. The highly integrated nature of these systems not only contributes to miniaturization through commonality, it allows fault tolerance and graceful degradation to be achieved with fewer resources.

MIREM is also applicable to conventional fault-tolerant systems in which fault tolerance is achieved through dedicated or "brute-force" redundancy. When applied to these systems, MIREM gives the same results as the well-known Reliability Block Diagram (RBD) analysis technique.

1.3 Evolution of the Model

The MIREM methodology was developed under the Impact Analysis of Integrated Communication, Navigation and Identification Avionics (ICNIA) program. The ICNIA system, which is currently in the advanced development phase under the direction of the Air Force Wright Aeronautical Laboratories (AFWAL), will integrate 16 radio functions into a highly reconfigurable system. The MIREM methodology was initially implemented in a PL/1 program and used to analyze the ICNIA system definition study architectures. This application demonstrated the utility of the methodology. System engineers from ICNIA contractor organizations found the model results to be reasonable and useful. Several additional features were requested by them and by AFWAL personnel.

Under a second phase, the methodology was enhanced to include the features that had been requested. The enhanced model was programmed in FORTRAN 77 and made much more usable by adding interactive data entry, input data checking and error messages, and improved output formats. An added benefit of the FORTRAN 77 program is its portability to a variety of computing environments. This version of MIREM was tested extensively against the PL/1 version and against hand-calculations. It has been used by Air Force personnel and contractors to analyze the ICNIA advanced development models.

A third version of MIREM, which this guide addresses, was created to respond to several requests for additional enhancements. The changes between this and the previous version of MIREM are listed in Table 1. These changes increase the accuracy of MIREM and make it useful in addressing logistics support

Table 1. New Features

Feature	Outputs Affected
Imperfect Switching (Undetected Failures and False Alarms)	New output option
Innovative Repair Policies and Availability	New output option
Graphical Output	All
Standby Redundancy	All
General Series-Parallel Structures (Groups)	All
Module Removal Rates	LRM/LRU Budget
Exact MCSP Algorithm	All except MTBFF
Size Changes	None
Increased Input Checking	None

issues encountered further into the development cycle, and during advanced development and full-scale development.

1.4 Organization of the Users Guide

A description of the reliability model is presented in Chapter 2. This description includes the rationale behind the model and its relationship to current fault-tolerant system reliability techniques. Detailed mathematical formulations are provided in Appendix A. Chapter 3 provides instructions on how to prepare model inputs from typical available data sources. User instructions for operating the program are given in Chapter 4. Sample studies and model outputs are presented in Chapter 5. Some advanced applications of the model that require more careful input data preparation are described in Chapter 6. A sample session listing is reproduced in Appendix B. Data entry forms for use in preparing model inputs are provided in Appendix C, and a glossary of model terminology is found in Appendix D.

2. RELIABILITY MODEL

The reliability model used in MIREM has two main distinguishing features:

1. The complex fault-tolerant structures found in dynamically reconfigurable systems are modeled.

2. The multiplicity of functions supported in highly integrated systems and the multifunction nature of individual hardware elements are modeled.

A network model is used to represent these system features in a reliability structure. Because it is network-based, MIREM is more economical than most simulation-based or Markov-based models. A new computational approach is used that allows realistic complexities in system configurations to be analyzed. In addition, a database that is organized by function is used, making the analysis of multifunction systems much more straightforward. These model characteristics make MIREM applicable to a broad class of advanced systems and necessitate the use of mission-related reliability measures (Sections 2.1 and 2.2). MIREM computations are described in Section 2.3. Finally, the relationship between MIREM analysis and the more traditional reliability block diagram approach is explored in Section 2.4.

2.1 Measures of Effectiveness

The multiplicity of functions supported by these systems and their varying importance to different operating scenarios necessitate a combined measure of effectiveness for reliability. It is assumed that the operating profile consists of a single "mission" that is repeated. The mission may contain several phases, each of which involves a set of required, or critical, functions. A system failure or critical failure, is defined as the unavailability of a critical function during a mission phase in which it is required. Using this definition of critical failures, reliability measures are defined as follows:

1. Mission Completion Success Probability (MCSP) - the probability that one mission is completed without a critical failure by a system that initially contains no faults.

2. Mean Time Between Critical Failure (MTBCF) - the mean operating time until a critical failure occurs, starting with a system that contains no faults.

3. Failure Resiliency - the ratio of MTBCF to the traditional Mean Time Between Failure (MTBF).

Since MTBF refers to the first failure in the system, failure resiliency is greater than or equal to 1 and can be roughly interpreted as the average number of failures until critical failure. Larger failure resiliency values correspond to systems with a higher degree of fault tolerance. The MCSP and MTBCF measures can also be defined for single functions instead of for mission scenarios. When a single function is being considered, MTBCF will be referred to as Mean Time Between Function Failure (MTBFF). The probability that a single function is available for a specified duration will be referred to as MCSP, and the function will be specified.

The following maintainability/availability measures will be used:

1. Mean Time Between Maintenance Action (MTBMA) - the mean operating time until system repair, starting with a fault-free system. Since system failure always causes a repair action, this number is less than or equal to MTBCF.

2. Mean Time To Repair (MTTR) - the mean time to repair the system by removing and replacing Line Replaceable Units (LRUs) or Line Replaceable Modules (LRMs).

3. Inherent Availability - the ratio of MTBMA to MTBMA plus MTTR; the fraction of time that the system is operational, neglecting any logistics delays.

4. Probability of Removal - the probability that an LRM/LRU will contain a fault (and therefore be removed) upon repair of the system.

Two other definitions are introduced concerning the relative impact of LRMs/LRUs:

1. Marginal MCSP - MCSP given that no failures occur in a specified LRM/LRU. This measure ranges from system MCSP for an LRM/LRU that makes no contribution to MCSP, up to 1 for an LRM/LRU that dominates MCSP.

2. Relative Contribution to MCSP - the probability that repair of a specified LRM/LRU could restore mission capability when a critical failure occurs. This measure ranges from 0 for an LRM/LRU that never causes critical failure, up to 1 for an LRM/LRU that dominates MCSP.

All of these measures depend on the repair policy being used; i.e., the decision of when to repair the system. Four policies will be considered:

1. Immediate Repair - repair any faults at the end of each mission.

2. Deferred Repair - repair only when a critical failure occurs.

3. Scheduled Maintenance - repair after a specified operating time or when a critical failure occurs.

4. Repair at Degraded Level - repair when the number of redundant components in some portion of the system falls below a specified level; these repairs include repairing when a critical failure occurs.

Not all of the performance measures can be computed for the last two repair policies, as explained in Section 4.7.

2.2 Mission Scenarios

A mission can be described by a time sequence of function requirements. MIREM assumes that the mission can be divided into phases, each of which has a set of critical functions. The ability of the system to support the requirements during a phase will also depend on the timing of function requirements within the phase. Because the specific timing is generally unknown, MIREM considers just two cases:

(a) All functions are required simultaneously within a phase.

(b) Each function is required independently within a phase.

These two cases bound the actual mission environment. The worst case (a) is recommended as the baseline for analysis.

2.3 MIREM Computational Approach

2.3.1 Resource Failure Model

From a reliability perspective, the system is considered to be a collection of discrete resources. A resource fails as a unit and is monitored individually by the system controller for reconfiguration purposes. Resources correspond to functional entities in the system. They often also correspond to

physical entities, such as LRMs. Switches, interconnections between LRMs or LRUs, BIT hardware, and control hardware can all be included as resources and their failures considered. Software failures per se are not included in MIREM; however, software characteristics that affect fault recovery and reconfigurability can be accounted for as described in Section 2.3.2.

All resources are assumed to have a constant failure rate, expressed as failures per operating hour, while they are activated. Each portion of the system (each pool) can be modeled as active redundant, with all resources activated and subject to failures whenever the system is operating, or as standby redundant, with resources activated only when they are being used to meet the mission requirement. To satisfy the constant failure rate assumption, individual resources should not contain redundancy.

2.3.2 System Failure Model

The basic computation performed in MIREM is to evaluate the probability of losing a specified functional capability ("system" or "critical" failure) over a specified operating time without repair; all other computations require this building block. A system failure is defined as occurring when there are insufficient resources to perform the required functions within a mission phase. This computation requires:

1. The resource failure model described in Section 2.3.1.

2. A mapping of resource failures into system failures.

Unfortunately, traditional approaches to evaluating this mapping are practical only for systems with a certain modular structure that does not always apply to advanced avionics architectures. Furthermore, it is desirable to represent this mapping for individual functions rather than complete missions, so that a variety of missions can be constructed from a single database.

For a broad class of advanced architectures, it is possible to take advantage of the special structure of this mapping to compute MCSP efficiently. The computations, as implemented in MIREM, are detailed in Appendix A. The basic approach is to assume a structure corresponding to two levels of reconfigurability or switching. This type of structure is illustrated in Figure 3.

At the lowest level, pools of interchangeable resources are identified. Branches are alternate identical paths within

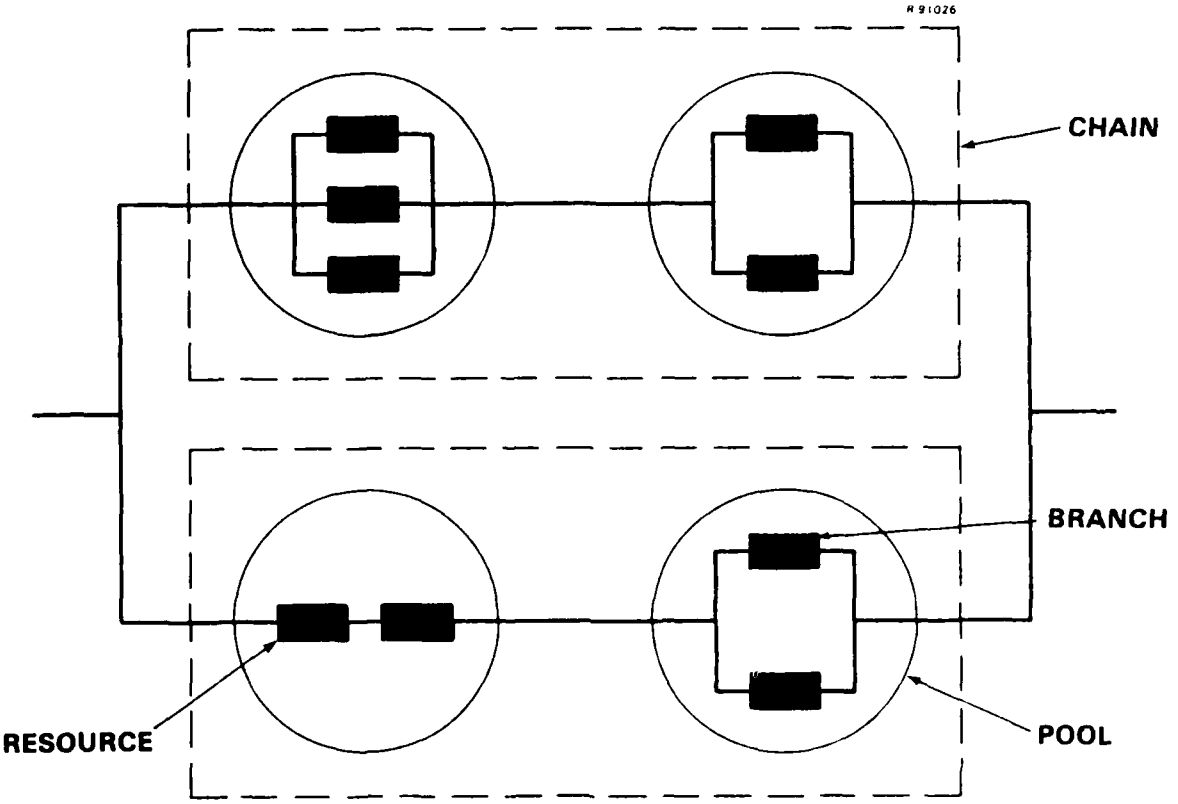


Figure 3. A Two-Level Structure for System Architecture Representation.

a pool, each containing one or more resources in series. Each function utilizes a certain number of branches (or fraction of a branch) in a pool. The combined resource requirement for a set of required functions depends on a number of timing issues and is addressed in Section 2.3.3. Given a total resource requirement of k, a pool with n parallel branches is evaluated as a k-of-n structure. MCSP for a set of series pools, called a chain, is the product of the probability that each pool has sufficient resources operating.

The second level of reconfiguration is between parallel chains. A chain is a set of pools that is switched (reconfigured) as a group. In many cases, a chain will correspond to an LRU because LRUs have separate power supplies and limited inter-LRU connections. A set of functions is available on parallel chains if there is an allocation of functions to chains such that each chain can support its allocated functions. The approach to evaluating MCSP on parallel chains

consists of enumerating all possible allocations of functions to chains (see Appendix A). This approach is computationally feasible whereas the traditional enumeration of resource states is not, the difference being that there are many more resources than required functions.

Communication between parallel chains is also modeled. For example, data processors in dual redundant LRUs may share their processing load through a data bus. This is modeled by defining shared pool pairs (one pool in each chain) that have a single resource requirement. The number of branches in this pool pair is the total for the two pools if both chains are operational; however, if certain other pools in a chain fail, its shared pools cannot be used.

Traditional series/parallel structures can also be analyzed using MIREM. In fact, if the multifunction capability of MIREM is not used, parallel chains are a series-parallel structure (actually a hierarchical k-of-n structure). This occurs when only one function is critical to a mission or when there is no contention between functions for resources (no type C or S pools, as defined in Section 2.3.3). However, using parallel chains restricts the model to two levels and two parallel paths at the higher level. More general structures, such as those shown in Figure 4, can be modeled in MIREM using groups, which are nested k-of-n structures.

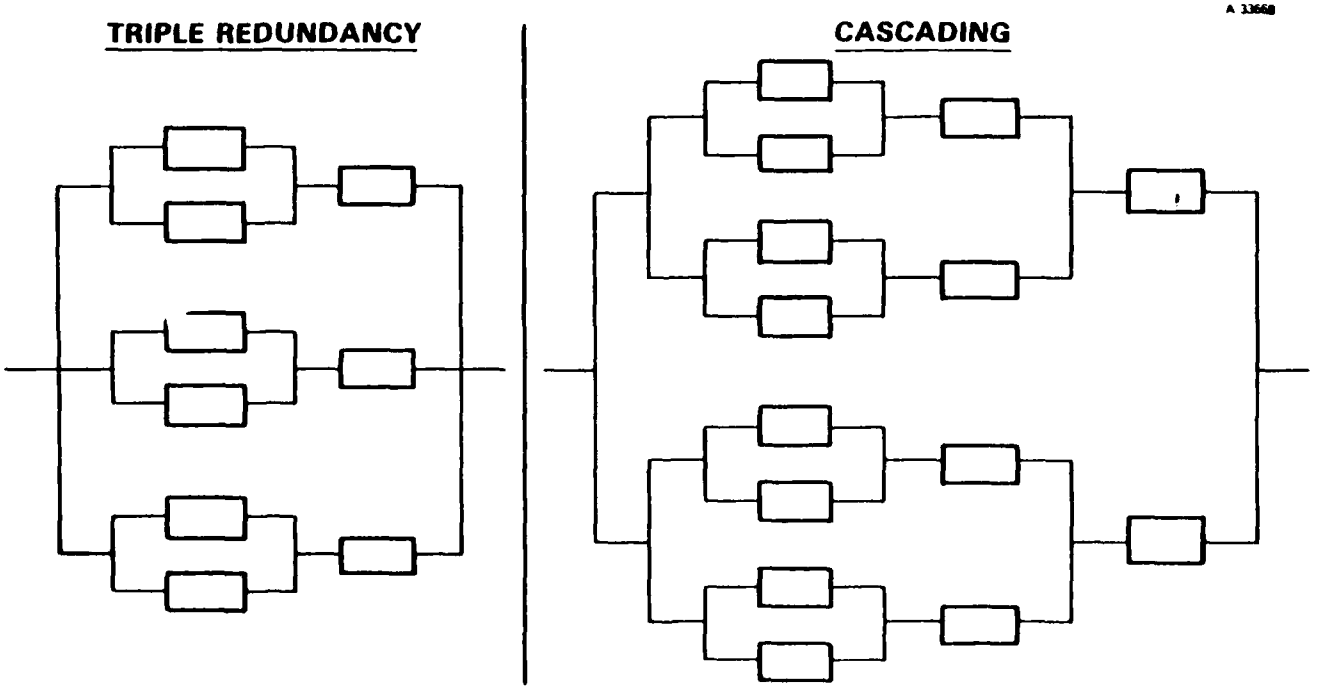


Figure 4. Examples of Series/Parallel Structures.

Total system MCSP is the product of the MCSP for each chain/parallel chain set. Other measures of effectiveness can be derived from MCSP. Of particular importance are: MTBCF, which is computed by evaluating and numerically integrating MCSP for different operating times; and failure resiliency, which is calculated as the ratio of MTBCF to MTBF.

2.3.3 Resource Requirements Model

A practical approach to determining the total resource requirements of functions that dynamically interact is to classify resources based on their dynamic features and then treat them accordingly in the network model described above. Three types of resource utilization have been identified:

1. Contending: Each function must use separate resources. The functions are available if separate resources are available for each function.

2. Timesharing: Each function utilizes a resource a fraction of the time. A set of functions is available if there is a configuration in which no resource is overloaded.

3. Noncontending: All functions can use the same resources. The functions are available if there are sufficient resources for the most demanding function.

Resources are contending with respect to certain functions if the resources must be dedicated constantly, or at rigidly scheduled times, to supporting the functions (e.g., receivers used to monitor communication channels). Resources are timeshared if they are utilized by a function at flexibly scheduled times so that several functions can be interleaved (e.g., data processors). Resources that can be used by any number of functions simultaneously (e.g., power supplies) are always noncontending.

The classification of resources as contending, noncontending, or timesharing also depends on the times during a mission phase at which each function is required. If functions are not required simultaneously, their resources are noncontending.

MIREM assumes that the type of resource utilization can be identified at the pool level; i.e., all resources within a pool are utilized in the same manner. Resource requirements are calculated according to four pool types that result from the resource utilization options:

N: noncontending pools, excluding type F

C: contending or timesharing pools, excluding type S

S: shared pools; i.e., contending or timesharing pool pairs in parallel chains that share requirements

F: chain-fail pools; i.e., noncontending pools that must be operating for any of the pools in the chain (including type S pools) to be used.

Resource requrements for type N and type F pools are the maximum of the individual function utilization rates. Resource requirements for type C and type S pools are the sum of the function utilizations for simultaneous mission requirements. When functions are not required simultaneously, the resource requirement is the maximum function utilization.

For pools in a parallel chain, the required functions are only those allocated to that chain. Type S pools, which always occur in pairs with one pool in each parallel chain, must support the functions allocated to both of the chains. Using these procedures, the number of branches (parallel resource paths) required in each pool is determined. These requirements can then be used to drive the system failure model described in Section 2.3.2.

2.3.4 Repair Model

The MCSP calculation of Section 2.3.2 applies to an operating period during which there is no repair. Noncritical failures accumulate until the system fails. MIREM also considers the various repair policies defined in Section 2.1. MCSP and maintainability calculations for different repair policies are discussed in this section. Because the deferral of repairs results in missions being started in various degraded (but still mission-capable) states, a single MCSP number does not apply. Instead, the average MCSP of a fleet of systems, operated under a repair policy until they have reached a steady-state distribution of system health, will be considered.

Immediate Repair - Under immediate repair, all missions are flown with a fully repaired system. The MCSP calculation of Section 2.3.2 gives the average MCSP. MCSP is converted to an average failure rate for a mission; the inverse of this failure rate is MTBCF. Since all failures are repaired, MTBMA is just MTBF.

Deferred Repair - Under deferred repair, repair corresponds to critical failure; i.e., MTBMA is equal to MTBCF. MTBCF is obtained by numerically integrating MCSP over different operating times without repair. Average MCSP is obtained from the average failure rate corresponding to MTBCF.

Scheduled Maintenance - Under scheduled maintenance, the system must operate for the scheduled maintenance interval without critical failure. MCSP is integrated over this interval and used to compute MTBCF. The calculation assumes that the scheduled maintenance "clock" is reset when a critical failure occurs. MTBMA is obtained by summing the rate of noncritical repairs at the scheduled times and the rate of critical repairs.

Repair at Degraded Level - Under this policy, repair occurs when the number of good branches in some pool drops below a specified minimum level of repair. Reliability measures are calculated only for systems with no parallel chains or groups. For series chains, average MCSP is calculated by first deriving the steady-state distribution of the number of branches remaining in each pool, using a Markov model. MCSP for each number of branches is then computed and the average taken. Pools are combined using an approximation that avoids having to model the joint state space, which would have too many states. MTBCF is again computed by converting average MCSP to an average failure rate.

MTBMA can be calculated for any system (including parallel chains) as the average time until some pool falls below its minimum repair level. It is obtained by numerical integration over time. For type S pool pairs, the minimum repair level is treated as the combined level for both pools. Repair occurs when either pool falls below half of the repair level. Because of this special treatment, the input repair level may be less than the mission requirement for type S pools (repair levels for other pool types must meet the mission requirements). Hence, MTBMA for parallel chains should be viewed as approximate. In particular, this calculation could give an MTBMA greater than the deferred repair MTBCF, which is incorrect.

2.3.5 Imperfect Switching Model

The previous sections have assumed that the system controller always selects a configuration that meets the mission requirements, if such a configuration exists. This assumption requires:

1. Perfect knowledge of faults from BIT.

2. Optimal reconfiguration logic in the controller.

MIREM contains an imperfect switching, or BIT, model that relaxes the first requirement by considering undetected failures and false alarms. Undetected failure rates and false alarm rates are specified for each pool. Mission outcome probabilities are computed for three cases:

1. Up and Believed Up - the critical functions were supported and the system controller believes they were supported.

2. Up and Believed Down - the critical functions were supported but the system controller believes they were not supported.

3. Down - the critical functions were not supported.

The probability that the system is up using the imperfect BIT model (the first two cases) is defined as the imperfect switching MCSP. The probability that the system is incorrectly believed to be down (the second case) is defined as the mission false abort probability.

Imperfect switching causes a critical failure when the controller selects a configuration in which a resource with an undetected failure is used, or when false alarms make the controller think that insufficient resources remain. In the latter case, it is assumed that a configuration will still be selected, but it may use resources with detected failures. The exact probabilities of these events depend on the specific reconfiguration logic used. MIREM calculates upper and lower bounds that apply to almost any reconfiguration logic.

For imperfect switching MCSP, the lower bound is obtained by assuming that any undetected failure will cause a critical failure; the upper bound, by assuming that the resources least prone to undetected failures will be used and neglecting reconfiguration due to failures. False alarms are neglected in both cases, making the lower bound approximate. Imperfect switching MCSP is then subtracted from perfect switching MCSP to show the contribution to mission failures. Numerical integration is performed to compute the imperfect switching MTBCF.

Two approaches are used to bound the mission false abort probability. The first is to subtract p_I, the probability

that the system is up and believed up, from the imperfect switching MCSP. Bounds for p_I are obtained by assuming that all components with undetected failures are used or that a minimum of them are used. False alarms are considered. Because these bounds can be very loose, a second upper bound is also computed using the approach described in Appendix A.8.

2.4 Reliability Block Diagram Interpretation of MIREM

MIREM considers structures (i.e., relationships between resource failures and system failure) that cannot be drawn exactly using RBDs. However, it can be very useful to represent portions, or configurations, of a MIREM structure using RBDs. Several RBD interpretations of MIREM are suggested. An example of a single function (and single pool) RBD is given in Chapter 3.

Single-Pool RBD - Each pool with pool type N, C, or F can be drawn as an active k-of-n RBD once the functions allocated to the chain using the pool are specified. The type S pool pairs can be drawn as a k-of-n RBD once the required functions are specified, assuming that certain other type F pools do not fail. *If they fail, the type S pool pair RBD changes.* The following rules define the RBD:

1. The resources within each branch of the pool are in series.

2. The number of parallel paths (n) is the total number of branches in the pool or pool pair.

3. The number of required paths (k) is the combined utilization rate of the pool across the functions using the pool or pool pair as defined in Section 2.3.3. Fractional utilizations are rounded upward.

Single-Function RBD - If only one function is required, an RBD can be drawn for the entire perfect switching system except that if the system contains type S and F pools, the type S pools cannot be interpreted exactly. The RBD suggested here neglects the interaction between type S and type F pools. To draw the RBD:

1. Form a k-of-n RBD for each pool or type S pool pair, assuming that the function is allocated to that pool's chain.

2. Form an RBD for each chain by placing the type N,C, and F pools in series. Some of the pools may have 0-of-n (irrelevant) RBDs for a given function and can be omitted.

3. Form an RBD for each parallel chain pair by placing the two chains in parallel (1-of-2) and then adding each type S pool pair RBD from these chains in series.

4. Form the system RBD by placing each single chain and parallel chain pair RBD in series.

Chain Allocation RBD - One level at which dynamic reconfiguration can occur in systems modeled by MIREM is the allocation of functions to parallel chains (dynamic reconfiguration also occurs within pools). If the configuration is restricted by assuming a certain allocation of functions to chains, a perfect switching system RBD can be drawn (disregarding the interaction between type S and type F pools). The RBD will depend on the required functions (the mission) and the chain allocation. To draw the RBD:

1. Form an RBD for each pool or type S pool pair based on the functions allocated to that pool's chain.

2. Form an RBD for each chain as above.

3. Form an RBD for each parallel chain pair by placing the two chains in series (2-of-2) and then adding each type S pool pair RBD from these chains in series. If a chain has no functions allocated to it, it can be omitted from the RBD.

4. Form the system RBD as above.

Series chains with groups can also be represented by RBDs once the required functions are specified. Each group is a k-of-n structure containing other groups or pools as its branches.

3. INPUT DATA PREPARATION

The reliability model used in MIREM requires as input a fault-tolerant system structure. Although the necessary information is generally available as soon as a top-level system design emerges, translating this information into a fault tree/block diagram/structure function can be a formidable task. The data structure approach used in MIREM simplifies this process. This chapter provides an orderly procedure for

identifying MIREM data elements. The process takes place off-line and results in data entry worksheets which can then be readily keyed in, as described in Chapter 4. Input data preparation for certain advanced applications that require approximate or case-by-case treatment are deferred until Chapter 6. The experienced user, when dealing with relatively simple architectures (or architecture changes), will probably be able to skip many of the off-line steps and proceed directly to on-line data entry; however, knowledge of this chapter will still be required.

3.1 Overview of Data Requirements

This section describes the data requirements for conducting an analysis with MIREM and introduces an example architecture which will be used throughout this manual. MIREM requires two basic types of data to define an architecture:

1. Structural data describing which system resources are required to perform each operational function and how these functions interact in their use of resources.

2. Reliability and maintainability data for each resource identified in the structural data.

Essentially, the structural data relate resource failures (point failures) to system failures for any specified mission environment. These data are typically available from several sources:

1. Function descriptions describe the signal processing path in terms of resources required to perform each function (Figure 5).

2. System and processing allocations describe how much of the system capacity (multiplexer, signal processor, or data processor throughput) is required to perform each function (Table 2).

3. System timing descriptions provide data on the times at which different functions utilize resources; i.e., whether functions can be scheduled to use resources at different times (resource sharing) or must use different resources at the same time (Figure 6).

GPS REQUIRES:	
1	L-BAND ANTENNA CONNECTOR
2	L-BAND RECEIVERS
1	2×3 L-BAND SWITCH
2	PREPROCESSORS
1	SIGNAL PROCESSOR
2	HIGH-SPEED DATA BUSES
1	POWER AND CONTROL
UHF REQUIRES:	
1	LOW-BAND ANTENNA SWITCH
1	LOW-BAND RECEIVER
1	2×5 LOW-BAND SWITCH
1	PREPROCESSOR
1	SIGNAL PROCESSOR
2	HIGH-SPEED DATA BUSES
1	POWER AND CONTROL
SINCGARS REQUIRES:	
	ALL UHF RESOURCES
1	SECURE DATA UNIT I/O

Figure 5. Function Descriptions for an Example Architecture.

Table 2. Processing Allocations for an Example Architecture

Function	Signal Processor Throughout (MIPS)[a]
GPS	8.0
UHF	1.0
SINCGARS	4.0

[a]Each signal processor has a capacity of 10 Million Instructions Per Second (MIPS).

Low-Band Receiver and Switches

- Can be shared (rapidly reprogrammed) between UHF and SINCGARS

Preprocessors

- Required continually by UHF and SINCGARS (cannot be rapidly reprogrammed)
- Required at fixed times by GPS

Signal Processors

- A function assigned to one Digital LRU can use the signal processor in the other LRU via the data buses
- Signal processors can only be powered by the power supply in their own LRU; however, they can receive control data from either controller

Power Supplies, Buses and Controllers

- Can handle all functions simultaneously

SDU I/O

- SINCGARS must use the SDU in the same LRU in which it is preprocessed for control reasons

Figure 6. Processing Descriptions for an Example Architecture.

4. Block diagrams show switching limitations that help to define the alternative signal processing paths that are available to perform each function (Figure 7).

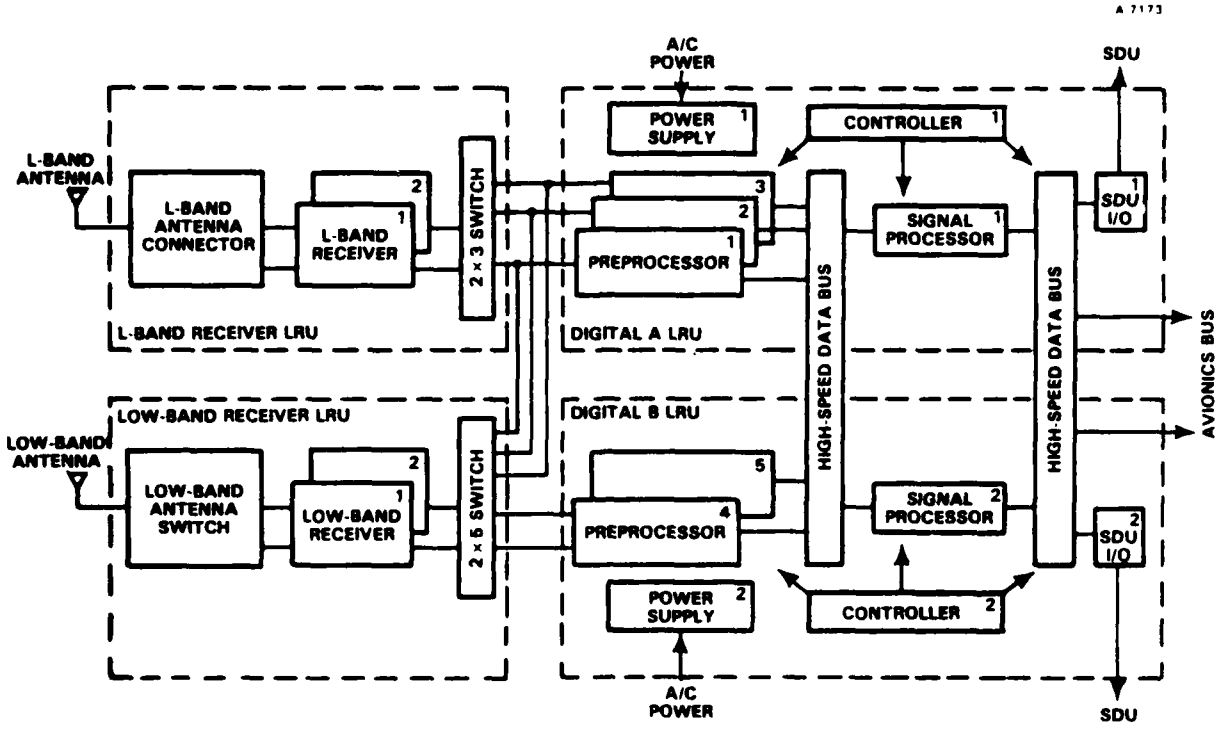


Figure 7. Block Diagram for an Example Architecture.

5. Software descriptions identify the fault recovery techniques used and their implication in terms of which faults can be recovered from (Figure 8).

- All resources use hot standby except for low-band receivers, which use cold standby

Controllers

- Either controller can control all functions in the event that the other controller fails

Signal processors

- The application program for each function is stored in both signal processors; either processor can take over a function if the other processor fails

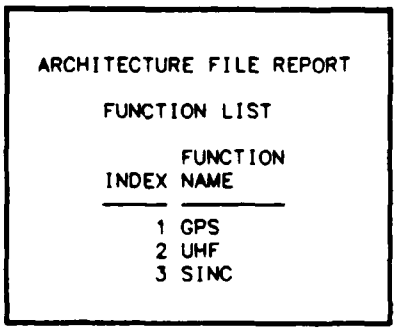
Figure 8. Fault Recovery Techniques for an Example Architecture.

Reliability and failure mode data, broken down to the level of resource failure rates, are generally available from standard reliability program tasks. If data on false alarm rates and undetected failure rates are available, they can also be incorporated in MIREM. Maintainability (MTTR) estimates can also be entered.

3.2 Preparing the Function List

The first step in preparing MIREM data for an architecture is to list the functions performed by the system at the operational or mission level. Functions are assigned sequential numbers, and names of eight characters or less (Figure 9).

Separate functions are needed for each system capability that may be stated as a separate requirement in a mission environment. Further separation of functions (e.g., separate receive and transmit) may be needed to simplify the description of resource requirements, if these functions can be allocated independently by the system controller.


ARCHITECTURE FILE REPORT

FUNCTION LIST

INDEX	FUNCTION NAME
1	GPS
2	UHF
3	SINC

Figure 9. Function List for the Example Architecture.

3.3 Preparing the Resource List

All elements of the system, including switches, control hardware, interconnections and external interfaces, are broken down into resources.

Each type of resource is assigned a

1. Resource number (up to three digits).

2. Quantity (the number of resources of this type in the system).

3. Failure rate (failures per million hours).

4. Classification as a "resource" (belonging to an LRM/LRU) or an "interconnection."

5. Mean Time to Repair (MTTR) (hours).

6. Resource name of 30 characters or less.

The resource list for the example architecture is shown in Figure 10. The switch matrices are broken down into their input ports, based on the assumption that loss of an input port is the primary failure mode. Data bus failures are not considered in this example.

ARCHITECTURE FILE REPORT

RESOURCE LIST

RESOURCE NUMBER	QUANTITY	FAILURE RATE (X E-6 HRS.)	RESOURCE/ INTERCONNECTION	MTTR (HOURS)	RESOURCE NAME
1	1	10	RESOURCE	4.0	L-BAND ANTENNA CONNECTOR
2	2	15	RESOURCE	3.5	L-BAND RECEIVER
3	2	5	RESOURCE	2.0	2 X 3 L-BAND SWITCH PORTS
4	1	10	RESOURCE	2.5	LOW-BAND ANTENNA SWITCH
5	2	95	RESOURCE	4.0	LOW-BAND RECEIVER
6	2	5	RESOURCE	3.0	2 X 5 LOW-BAND SWITCH PORTS
7	5	300	RESOURCE	2.0	PREPROCESSOR
8	2	100	RESOURCE	5.0	SIGNAL PROCESSOR
9	2	20	RESOURCE	4.5	POWER SUPPLY
10	2	20	RESOURCE	4.0	SDU I/O
11	2	100	RESOURCE	6.0	CONTROLLER

Figure 10. Resource List for the Example Architecture.

Resources are defined as functional entities that:

1. Do not contain fault tolerance; i.e., they fail as a unit.

2. Are monitored and switched by the system controller as a single unit.

3. If used by multiple functions, cannot support any of these functions after a failure occurs.

This definition sets the level of detail at which a MIREM analysis should be conducted. A higher-level view of the system will not capture all of the fault tolerance inherent in the design. If a redundant structure is modeled as a single resource, the effect of its fault tolerance is lost. If two specialized resources that are used by different functions are treated as a single resource, the ability of each function to tolerate faults will be understated. In practice, it may not be possible to break out all resources according to this definition. A higher-level analysis can provide useful insights; however, the unavailability of these data should raise questions as to how well the fault tolerance issues have been addressed.

Although resources are defined as functional entities, they often correspond to physical units. In current avionics packaging technology, resources often correspond to Shop Replaceable Units (SRUs). In advanced modular packaging, they often correspond to LRMs. However, MIREM is not limited to these physical levels of redundancy. Redundancy within a module or even within a chip can be modeled by defining resources appropriately.

3.4 Identifying Resource Chains

The process of identifying the fault-tolerant structure of an architecture begins at the top level with resource chains. Chains can be identified by examining the system block diagram; parallel chains correspond to the highest level of switching or redundancy in the system (see Figure 3) and series chains contain all resources that are not redundant at this level. In the example architecture (Figure 11), the dual redundant digital LRUs form two chains in parallel because the low-band functions can be routed through either LRU. The fact that the two LRUs are connected by data buses does not prevent them from being parallel chains (these are treated in Section 3.5); note that functions can cross to the other LRU for the signal processor only. All other resources (i.e., both receiver LRUs) form a single series chain. Although these LRUs contain redundancy, they can be modeled as a series chain because they contain only one level of redundancy. In general, a series chain must contain no more than one level of redundancy; parallel chains are used to model two levels of redundancy.

Another restriction is that MIREM does not allow more than two chains in parallel. However, large numbers of parallel paths can be modeled at the lower level of redundancy in the two-level MIREM structure, the pool level. For more redundant paths or more levels of redundancy, the group feature of Section 3.6 can be used. The number of series chains used does

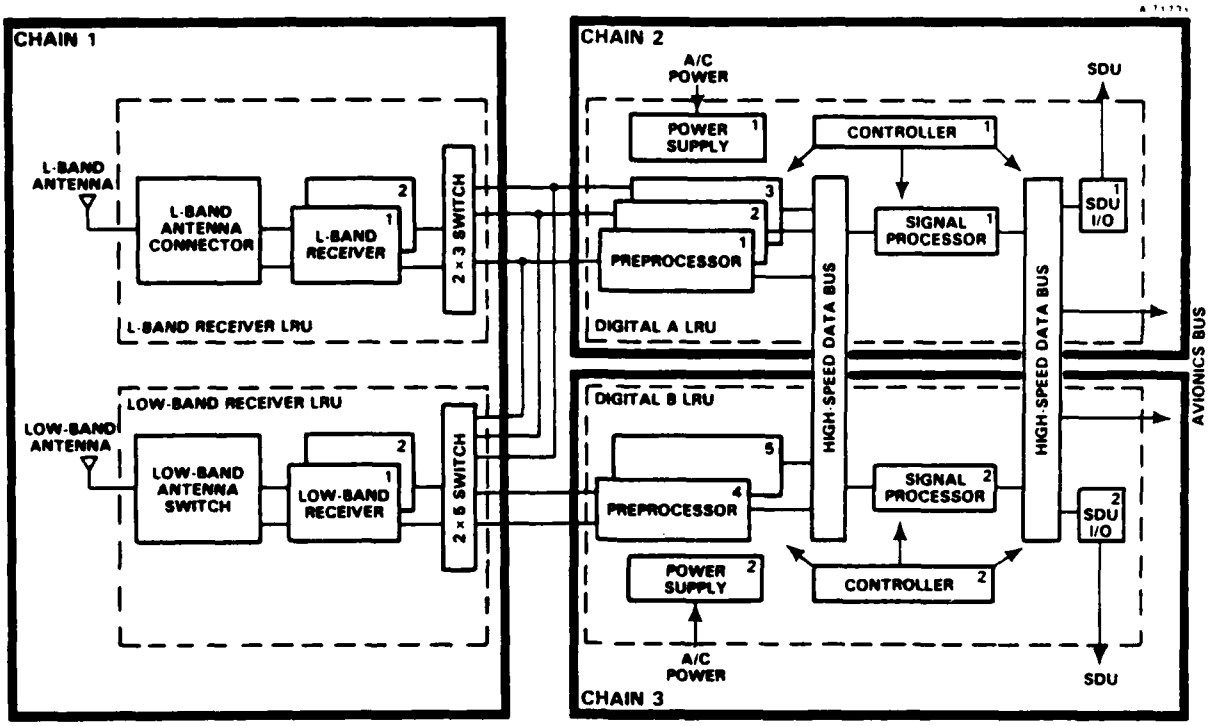


Figure 11. Resource Chains for the Example Architecture.

not affect the system-level results; however, several chains may be desirable because the reliability budget is presented by chain. The criteria for identifying resource chains are summarized in Table 3.

At this point the function descriptions (Figure 5) are used to determine which functions can be processed in each chain. These function numbers are listed for each chain as shown in Figure 12. Each chain is assigned a number. One entry is made for each series chain or parallel chain pair (for convenience, each entry will be referred to as a chain pair). Each chain pair is assigned a name of 30 characters or less.

3.5 Identifying Resource Pools

The next, and most complex, step in preparing the input data is to divide the resource chains into pools. A pool consists of one or more identical branches in parallel; each branch contains one or more resources (see Figure 3). In some cases pools are easily identified. Several data processors that are connected by data buses, share processing loads, and

Table 3. Criteria for Identifying Resource Chains

Type of Chain	Criteria
Parallel	The highest-level switching points or redundancy in the system correspond to chain boundaries. A function may use only one of the parallel chains at a time (but functions may "cross over" to use certain resources in another chain; see Section 3.5) Only two chains are allowed in parallel LRUs often correspond to chains
Series	All resources not in parallel chains form a series chain Functions must use a series chain (no alternate path) Series chains contain no more than one level of redundancy: i.e., each parallel path in the chain must be a simple series structure, not containing redundancy

ARCHITECTURE FILE REPORT

CHAIN LIST

CHAIN PAIR NAME	CHAIN NUMBER	FUNCTIONS REQUIRED
FRONT END	1	1,2,3
DIGITAL	2	1,2,3
	3	2,3

Figure 12. Chain Data for the Example Architecture.

have fault recovery capability (each can resume any application program being run on another processor) clearly form a pool. A bank of fully switched identical receiver channels is another example. When pool boundaries are not obvious, the following approach is recommended.

Prepare a resource utilization worksheet that shows how each set of resources is utilized by each function. A partial worksheet for the example architecture is shown in Table 4. Each entry in the worksheet describes how a set of resources is used by functions, based on the function descriptions (Figure 5) and other data. Criteria for identifying these entries, or candidate resource pools, are listed in Table 5. Pool examples are shown in Figure 3. One additional criterion applies to worksheet entries (but not to pools) - structures with only one branch should contain only one resource; these resources are in series with the rest of the chain. The structure is recorded in terms of the number of parallel branches and the resources on each branch (if branches differ, see Section 6.1). Make a single entry for each structure, listing the number of branches required for each function. However, if different functions use the same resources in different structures, make duplicate entries for these resources (see Section 6.1).

Assign pool types (see Sections 2.3.2 and 2.3.3) to the entries as shown in Table 6. In Table 4, entries 10 and 11 are labelled type N because they are used only by the Global Positioning System (GPS); similarly, entry 24 is used only by the Single-Channel Ground and Airborne Radio Subsystem (SINCGARS). Entries 12, 13, and 23 are type N because they are used in a noncontending fashion (see Figure 6); similarly, entries 20 and 30 are type C because they are used in a contending fashion. Entry 21 and its counterpart in chain 3 are type S because the signal processors share loads. Entry 22 is type F because a signal processor must use the power supply in its own LRU.

Map the candidate pool entries into resource pools using the following procedures:

1. Entries in a chain with the same pool type (N or F), the same functions, and no redundancy are combined to form one pool.

2. All other entries map one-to-one into pools (but see Section 6.1 if multiple entries have been used for the same resources or if type C pools in parallel chains do not occur in pairs).

Record these pools on pool data entry forms (Appendix C) to facilitate on-line data entry, by conducting the following steps:

Table 4. Resource Utilization Worksheet

Entry Number	Pool[a] Type	Number of Branches	Resources	Functions
Chain 1 - Front End				
10	N	1	L-Band Ant. Conn.	GPS(1)
11	N	2	L-Band Receiver 2 x 3 Switch Port	GPS(2)
12	N	1	Low-Band Ant. Switch	UHF(1) SINC(1)
13	N	2	Low-Band Receiver 2 x 5 Switch Port	UHF(1)
Chain 2 - Digital A				
20	C	3	Preprocessor	GPS(2) UHF(1) SINC(1)
21	S	1	Signal Processor	GPS(1) UHF(1) SINC(1)
22	F	1	Power Supply	GPS(1) UHF(1) SINC(1)
23	N	1	Controller	GPS(1) UHF(1) SINC(1)
24	N	1	SDU I/O	SINC(1)
Chain 3 - Digital B				
30	C	2	Preprocessor	UHF(1) SINC(1)

[a]N - Noncontending, C - Contending, S - Shared, F - Chain-Fail.

Table 5. Criteria for Identifying Resource Pools[a]

Pool/Structure Characteristic	Criteria
Branch	Contains resources in series (no redundancy within a branch) Each resource in a branch can be used by the same set of functions Branches do not cross reconfiguration (switching) points
Number of Branches	Each branch can be used by the same set of functions All branches (i.e., the whole pool/structure) must be in one chain

[a]These criteria also apply to identifying entries in the resource utilization worksheet.

1. Assign unique pool numbers to each pool except for type C and S pairs, which must be assigned the same number.

2. Obtain the number of the chain containing each pool from the resource utilization worksheet.

3. Assign the LRM/LRU index, as discussed in Section 3.6.

4. Label the pool "active" if active redundancy is required (or if the pool has no redundancy); label it "standby" if standby redundancy is allowed.

5. Look up resource type numbers in the resource list.

6. Assign undetected failure rates and false alarm rates as fractions of all the failures in the pool (pool failure rates can be determined by summing the failure rates of resources in the pool from the resource list).

7. Assign utilization rates, as described below.

8. If repair policies are being considered, assign the minimum numbers of branches that must be operating in order for the system to be allowed to perform another mission without maintenance.

Table 6. Criteria for Assigning Pool Types

Pool Type	Characteristic
S	Entries/pools occur in pairs, one on each chain of a parallel chain pair Pair of entries/pools shares requirements (e.g., processors connected by a data bus) Pair of entries/pools has the same branch failure rate Dependent upon certain other resources in the chain
F	Occur in parallel chain pairs with type S pools Upon failure, would prevent all resources in the chain, including type S, from being used
N	All entries/pools that are not type S or F and are used only by one function Entries/pools that are not type S or F and are used in a noncontending fashion (see Section 2.3.3)
C	Entries/pools that are not type S or F and are used in a contending or timesharing fashion (see Section 2.3.3)

A completed pool data entry form for the example architecture is shown in Figure 13. Entries 10, 11, and 12 have been combined to form pool 10 because they contain no redundancy. Although entry 11 has two branches, both are required for GPS. It may appear that entries 22, 23, and 24 could be combined to form a series structure; however, if they were combined, their distinctive pool types and functions would be lost (e.g., the SDU I/O would become a critical failure for GPS and SINCGARS).

Figure 13 also includes the utilization rate in each pool for each function. For most pools, the utilization rate is

POOL DATA SHEET

A 16380a

Date: 3/4/86 Page 1 of 3

System: MIREM 3 Test #1 Architecture File Name: TEST1 Analyst: M Veatch

POOL	POOL NO.	CHAIN NO: 1	LRU INDEX: 1	POOL TYPE: N	NO. BRANCHES: 1	ACTIVE/STANDBY: A	UNDETECTED FAILURES: 0.10	FALSE ALARMS: 0.05	MIN REPAIR LEVEL: 1
POOL RESOURCE	10	1, 2, 2, 3, 3							
POOL UTIL		1, 0, 0							

POOL	POOL NO	CHAIN NO: 1	LRU INDEX: 1	POOL TYPE: N	NO. BRANCHES: 1	ACTIVE/STANDBY: A	UNDETECTED FAILURES: 0.05	FALSE ALARMS: 0.001	MIN REPAIR LEVEL: 1
POOL RESOURCE	11	4							
POOL UTIL		0, 1, 1							

POOL	POOL NO	CHAIN NO: 1	LRU INDEX: 1	POOL TYPE: N	NO. BRANCHES: 2	ACTIVE/STANDBY: S	UNDETECTED FAILURES: 0.02	FALSE ALARMS: 0.01	MIN REPAIR LEVEL: 1
POOL RESOURCE	12	5, 6							
POOL UTIL		0, 1, 1							

POOL	POOL NO.	CHAIN NO: 2	LRU INDEX: 2	POOL TYPE: C	NO. BRANCHES: 3	ACTIVE/STANDBY: A	UNDETECTED FAILURES: 0.01	FALSE ALARMS 0.01	MIN REPAIR LEVEL 2
POOL RESOURCE	13	7							
POOL UTIL		2, 1, 1							

POOL	POOL NO.	CHAIN NO: 2	LRU INDEX: 2	POOL TYPE: S	NO. BRANCHES: 1	ACTIVE/STANDBY: A	UNDETECTED FAILURES: 0.2	FALSE ALARMS: 0.05	MIN REPAIR LEVEL: 1
POOL RESOURCE	14	8							
POOL UTIL		.8, .1, .4							

POOL	POOL NO.	CHAIN NO: 1	LRU INDEX: 1	POOL TYPE: F	NO. BRANCHES: 1	ACTIVE/STANDBY: A	UNDETECTED FAILURES: 0	FALSE ALARMS: 0.01	MIN REPAIR LEVEL: 1
POOL RESOURCE	15	9							
POOL UTIL		1, 1, 1							

Figure 13. Pool Data Entry Form for the Example Architecture.

POOL DATA SHEET

A 15300

Date: 3/4/86 Page 2 of 3

System: MIDEM3 [illegible] #1 Architecture File Name: TEST1 Analyst: M. Veatch

POOL	POOL NO	CHAIN NO: 2	LRU INDEX: 2	POOL TYPE: N	NO. BRANCHES: 1	ACTIVE/STANDBY: A	UNDETECTED FAILURES: 0.01	FALSE ALARMS: 0.005	MIN REPAIR LEVEL: 1
POOL RESOURCE	16	10							
POOL UTIL		0, 1							

POOL	POOL NO	CHAIN NO: 2	LRU INDEX: 2	POOL TYPE: N	NO. BRANCHES: 1	ACTIVE/STANDBY: A	UNDETECTED FAILURES: 0.05	FALSE ALARMS: 0.02	MIN REPAIR LEVEL: 1
POOL RESOURCE	17	11							
POOL UTIL		1, 1, 1							

POOL	POOL NO	CHAIN NO: 3	LRU INDEX: 3	POOL TYPE: C	NO. BRANCHES: 2	ACTIVE/STANDBY: A	UNDETECTED FAILURES: 0.1	FALSE ALARMS: 0.01	MIN REPAIR LEVEL: 2
POOL RESOURCE	13	7							
POOL UTIL		2, 1, 1							

POOL	POOL NO	CHAIN NO: 3	LRU INDEX: 3	POOL TYPE: S	NO. BRANCHES: 1	ACTIVE/STANDBY: A	UNDETECTED FAILURES: 0.02	FALSE ALARMS: 0.015	MIN REPAIR LEVEL: 1
POOL RESOURCE	14	8							
POOL UTIL		.8, .1, .4							

POOL	POOL NO	CHAIN NO: 3	LRU INDEX: 3	POOL TYPE: F	NO. BRANCHES: 1	ACTIVE/STANDBY: A	UNDETECTED FAILURES: [illegible]	FALSE ALARMS: [illegible]	MIN REPAIR LEVEL: 1
POOL RESOURCE	15	9							
POOL UTIL		1, 1, 1							

POOL	POOL NO	CHAIN NO: 3	LRU INDEX: 3	POOL TYPE: N	NO. BRANCHES: 1	ACTIVE/STANDBY: A	UNDETECTED FAILURES: 0.01	FALSE ALARMS: 0.015	MIN REPAIR LEVEL: 1
POOL RESOURCE	16	10							
POOL UTIL		0, 0, 1							

Figure 13. (Continued).

POOL DATA SHEET

A 16,388a

Date: 3/4/86 Page 3 of 3

System: MIREM 3 Test #1 Architecture File Name: TEST1 Analyst: M. Veatch

POOL	POOL NO	CHAIN NO: 3	LRU INDEX: 3	POOL TYPE: N	NO. BRANCHES: 1	ACTIVE/STANDBY: A UNDETECTED FAILURES: 0.05	FALSE ALARMS: 0.02	MIN REPAIR LEVEL: 1
POOL RESOURCE	17	11						
POOL UTIL		1 1 1						

POOL	POOL NO	CHAIN NO:	LRU INDEX:	POOL TYPE:	NO. BRANCHES:	ACTIVE/STANDBY: UNDETECTED FAILURES:	FALSE ALARMS:	MIN REPAIR LEVEL:
POOL RESOURCE								
POOL UTIL								

POOL	POOL NO	CHAIN NO:	LRU INDEX:	POOL TYPE:	NO. BRANCHES:	ACTIVE/STANDBY: UNDETECTED FAILURES:	FALSE ALARMS:	MIN REPAIR LEVEL:
POOL RESOURCE								
POOL UTIL								

POOL	POOL NO	CHAIN NO:	LRU INDEX:	POOL TYPE:	NO. BRANCHES:	ACTIVE/STANDBY: UNDETECTED FAILURES:	FALSE ALARMS:	MIN REPAIR LEVEL
POOL RESOURCE								
POOL UTIL								

POOL	POOL NO	CHAIN NO:	LRU INDEX:	POOL TYPE:	NO. BRANCHES:	ACTIVE/STANDBY: UNDETECTED FAILURES:	FALSE ALARMS:	MIN REPAIR LEVEL:
POOL RESOURCE								
POOL UTIL								

POOL	POOL NO	CHAIN NO:	LRU INDEX:	POOL TYPE:	NO. BRANCHES:	ACTIVE/STANDBY: UNDETECTED FAILURES:	FALSE ALARMS:	MIN REPAIR LEVEL:
POOL RESOURCE								
POOL UTIL								

Figure 13. (Concluded).

merely the number of branches required by each function, taken from the resource utilization worksheet. However, for time-sharing resources such as processors and multiplexers, which can support several functions up to some maximum capacity, the utilization rate is the fraction of the capacity that is required to perform the function. The signal processors in pool 14 have a throughput capacity of 10 MIPS. Since GPS requires eight MIPS (Table 2), its utilization is 0.8. Pool pairs are required to have the same utilization for functions that can use both pools.

This completes the preparation of the structural data for this example. The pool and chain structure that has been created for the example architecture is illustrated in Figure 14. The next section describes an alternate data format that can be used for series-parallel structures (i.e., when there is no contention between functions).

3.6 Identifying Resource Groups

Architectures with more than two levels of redundancy will not fit the pool and chain structure described in Sections 3.3 and 3.4. If they do not exhibit contention between functions (i.e., there is only one function or all pools are type N) these architectures can usually be modeled as series-parallel structures; MIREM allows these structures to be defined using groups.

The example structure in Figure 15 could not be modeled using parallel chains because it contains three levels of redundancy. The figure illustrates how the structure is represented by groups, with each group containing pools or other groups. The data required for a group are a subset of the data required for a pool:

1. A unique group number; group numbers must be between 1000 and 1999, whereas pool numbers must be from 1 to 999.

2. The chain number.

3. The LRM/LRU index.

4. The group type (analogous to pool type); only type N is allowed.

5. The number of pools or groups contained in the group.

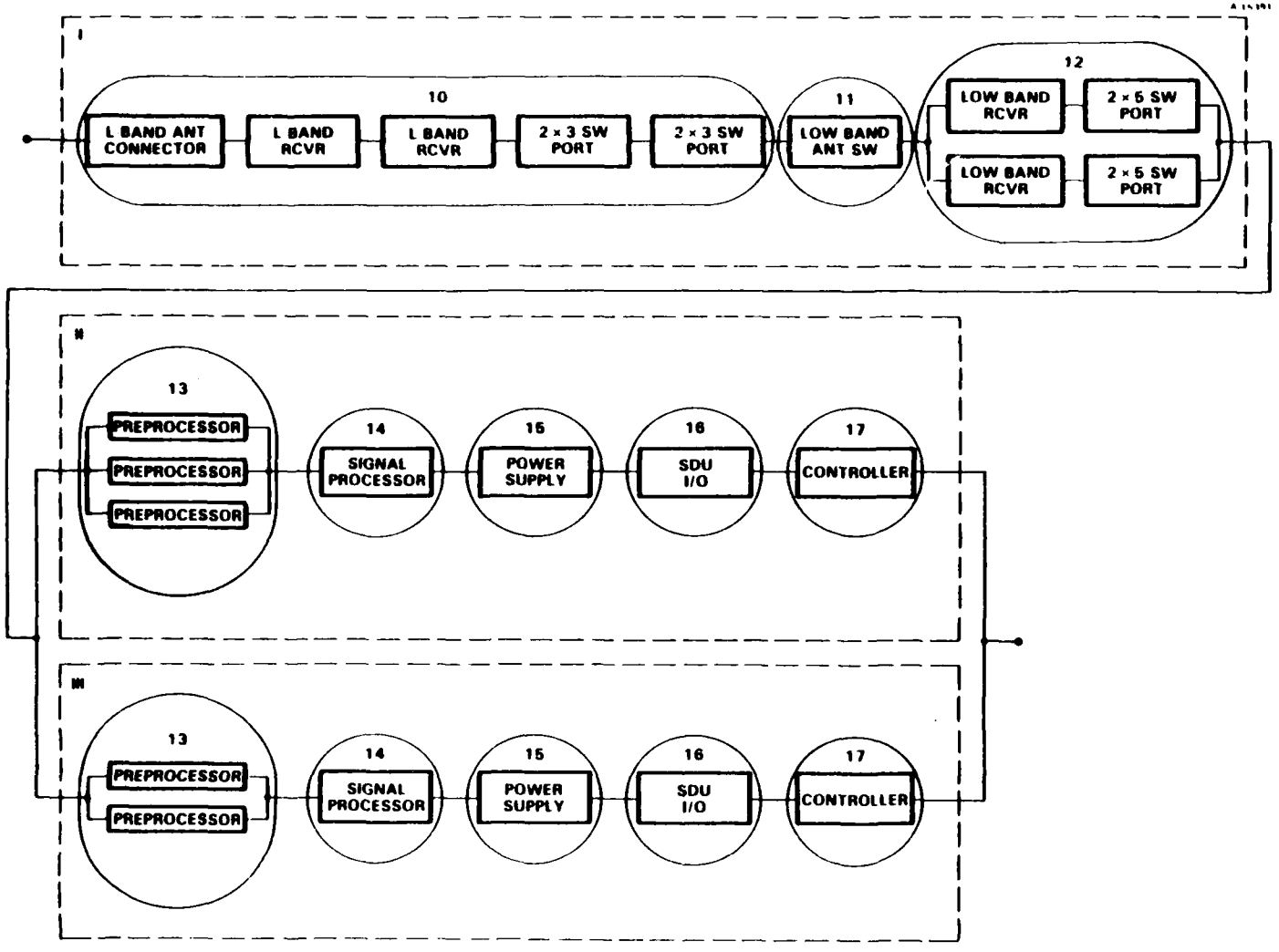


Figure 14. Pool and Chain Structure for the Example Architecture.

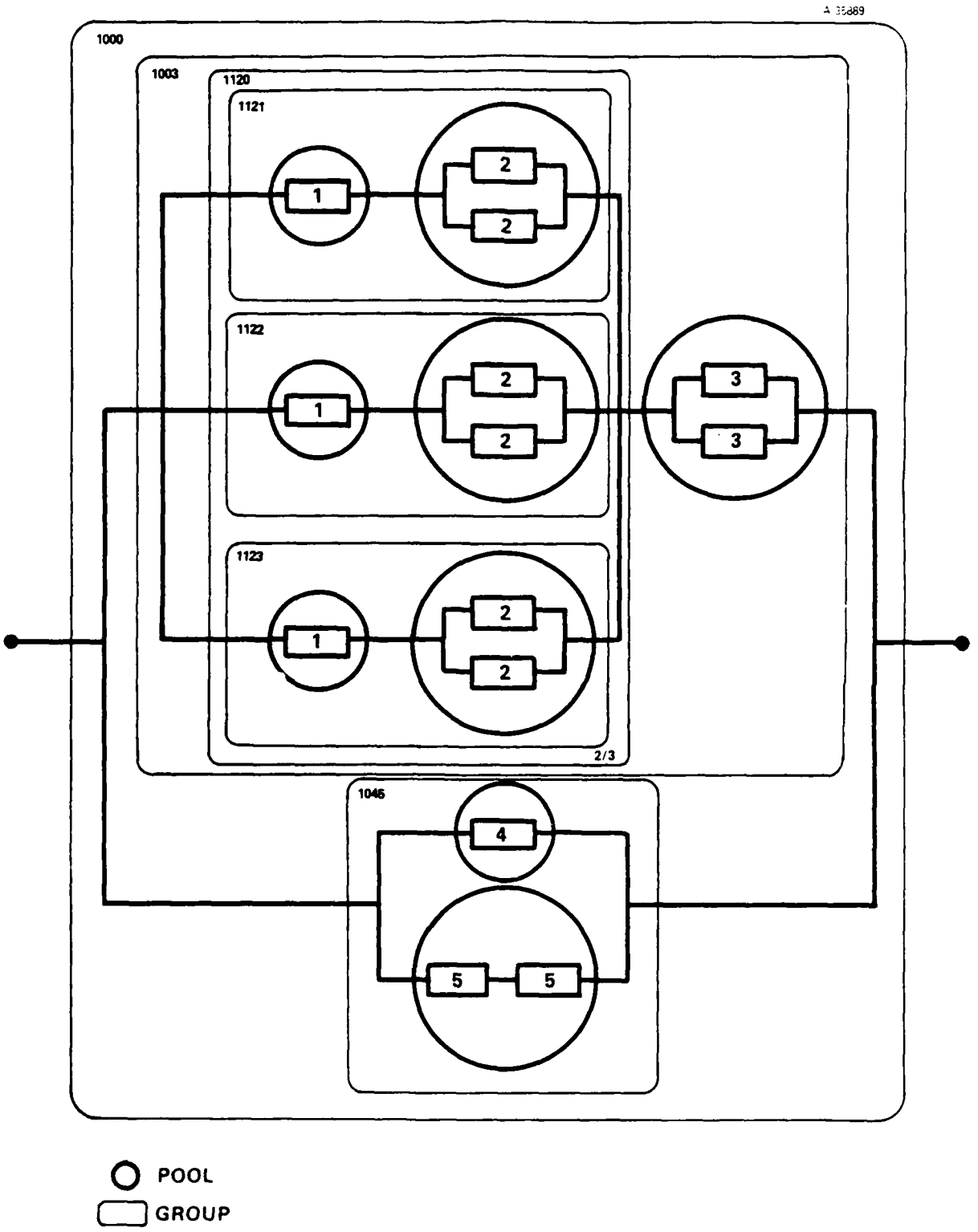


Figure 15. Example Series-Parallel Structure.

6. The group "resources"; i.e., the list of pools and groups contained in the group.

7. The "utilization" of the group by each function; i.e., the number of "resources" required.

The pool and group data for Figure 15, assuming only one function, are shown in Figure 16. Groups must be ordered so that all pools or groups contained in a group precede it in the input file. The last group is assumed to represent the entire chain.

ARCHITECTURE FILE REPORT

POOL REPORT

CHAIN NUMBER	POOL NUMBER	LRM/LRU NAME	NUMBER BRANCHES	POOL FAILURE RATE •	POOL TYPE	REDUN-DANCY	MINIMUM LEVEL REPAIR	UNDETECTED FAILURE RATE	FALSE ALARM RATE	RESOURCE NUMBER	RESOURCE FAILURE RATE •	RESOURCE NAME
1	11	L1	1	1000	NONCONTENDING	ACTIVE	0	0.000	0.000	1	1000	RES1
	12	L1	1	1000	NONCONTENDING	ACTIVE	0	0.000	0.000	1	1000	RES1
	13	L1	1	1000	NONCONTENDING	ACTIVE	0	0.000	0.000	1	1000	RES1
	21	L1	2	2000	NONCONTENDING	ACTIVE	0	0.000	0.000	2	1000	RES2
	22	L1	2	2000	NONCONTENDING	ACTIVE	0	0.000	0.000	2	1000	RES2
	23	L1	2	2000	NONCONTENDING	ACTIVE	0	0.000	0.000	2	1000	RES2
	31	L1	2	2000	NONCONTENDING	ACTIVE	0	0.000	0.000	3	1000	RES3
	41	L1	1	1000	NONCONTENDING	ACTIVE	0	0.000	0.000	4	1000	RES4
	51	L1	1	2000	NONCONTENDING	ACTIVE	0	0.000	0.000	5	1000	RES5
										5	1000	RES5
SYSTEM FAILURE RATE				14000								

(•) FAILURE RATE IN PER MILLION HOURS

ARCHITECTURE FILE REPORT

SUBGROUP LISTING BY GROUP

GROUP NUMBER	CHAIN NUMBER	GROUP TYPE	POOL/GROUP LIST
1121	1	N	11, 21,
1122	1	N	12, 22,
1123	1	N	13, 23,
1120	1	N	1121,1122,1123,
1003	1	N	1120, 31,
1045	1	N	41, 51,
1000	1	N	1003,1045,

Figure 16. Pool and Group Data for Figure 15.

3.7 Preparing the LRM/LRU List

The final step in preparing the input data for an architecture is to divide the resource pools into sets to be used for generating a reliability budget (this step is optional). One useful partitioning is to divide the system into removable units, namely LRUs or LRMs. A pool may contain several LRMs of the same type; therefore, it is recommended that all LRMs of the same type be included in one set. Consecutive numbers are assigned to the LRM/LRU groups, as shown in Figure 17 for

INDEX	LRM/LRU NAME
1	FRONTEND
2	DIGITALA
3	DIGITALB

Figure 17. LRM/LRU List for the Example Architecture.

the example architecture. These LRM/LRU indexes are then entered for each pool on the pool data entry form (Figure 13).

3.8 Using Reliability Block Diagrams

A Reliability Block Diagram (RBD) for each function, which relates to the MIREM structure as described in Section 2.5, can be used in place of function descriptions and block diagrams in the MIREM data preparation process. The RBD for the Ultra-High Frequency (UHF) voice communication function in the example architecture is shown in Figure 18. These diagrams can be used to fill out the resource utilization worksheet. Simple k-of-n structures (each function requires a different number of branches k) and single resources in series with these structures form the candidate pools. Nested parallel structures form parallel chains (for more than two levels of nested parallel structures or more than two parallel paths at the higher level, resource groups must be used). It should be noted, however, that type S and type F resources cannot be exactly represented in an RBD. The signal processors in Figure 18 are actually dependent on the power supplies. This information must be obtained from other sources in order to model them correctly as type S and type F pools, respectively. Additional information on the contention between functions is also required to determine pool type and determine the total number of branches required in each pool.

4. PROGRAM OPERATING PROCEDURES

4.1 Program Overview

To exercise MIREM the user interacts with three programs, as shown in Figure 19. Program DATAIN is used to create, review, and modify the two types of files required to operate

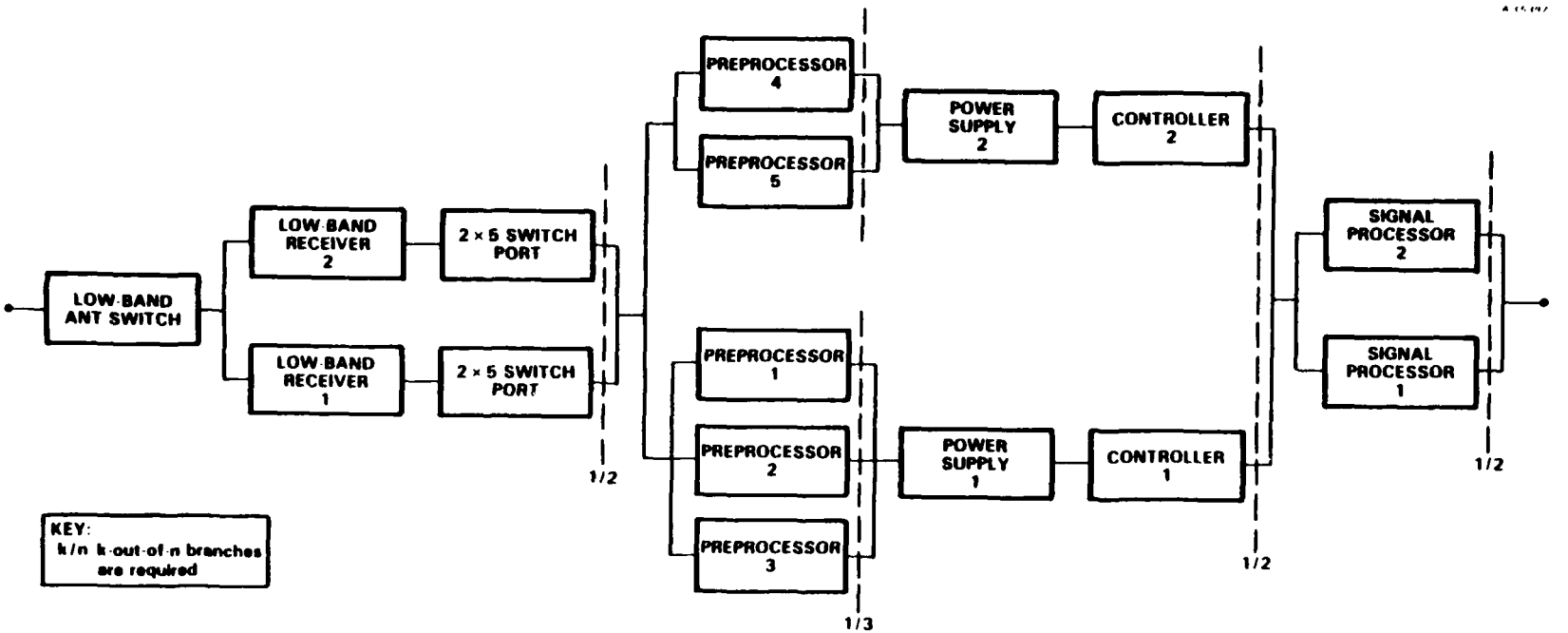


Figure 18. UHF RBD for the Example Architecture.

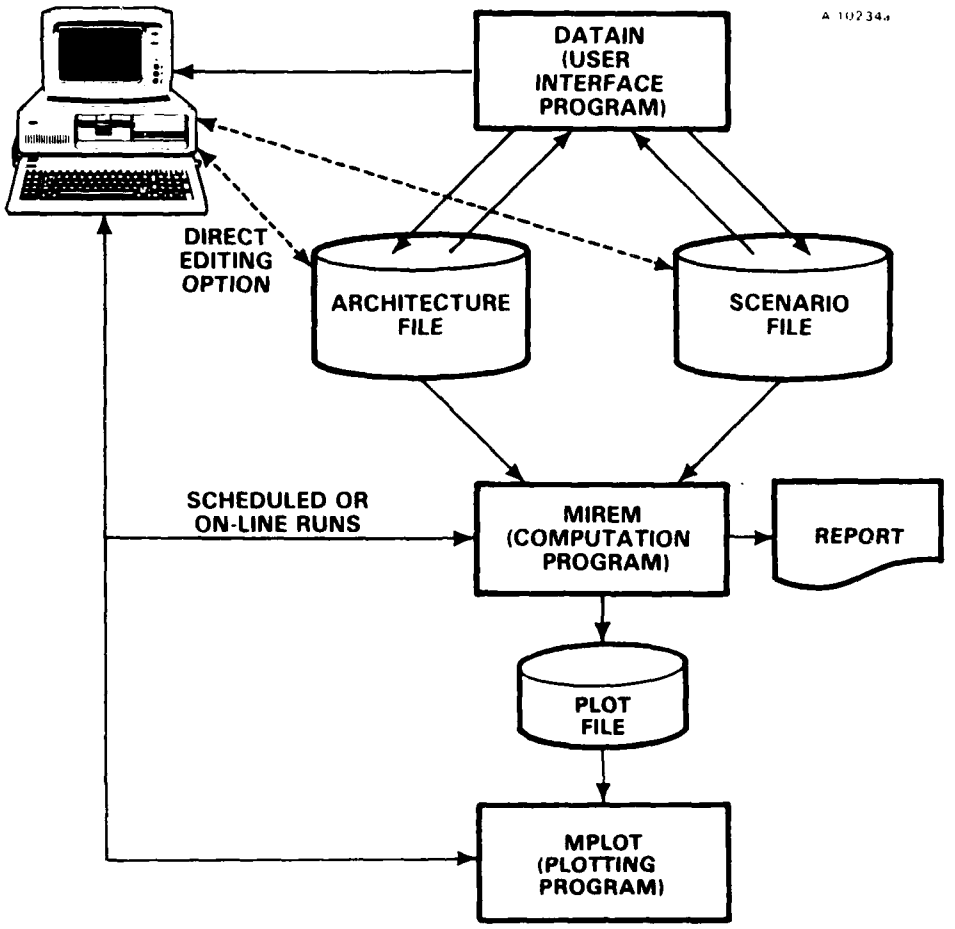


Figure 19. MIREM Program Overview.

MIREM. Program MIREM is used to generate reports on the contents of these files and the model results, and to create plot files. Program MPLOT is used to display graphical results stored in the plot files. A direct editing option is also provided that allows the advanced user to enter changes directly to architecture and scenario files rather than through DATAIN.

The programs are written in FORTRAN 77 and can be operated in a wide variety of computing environments. Program DATAIN operates interactively and is compatible with most full-screen and scrolling terminals. DATAIN can also be used, though less quickly, with hard-copy terminals. Program MPLOT can be used in a wide variety of interactive environments but requires the DI3000 graphics package and associated device drivers.

This chapter contains procedures for operation of programs DATAIN (Section 4.2), MIREM (Section 4.4), and MPLOT (Section 4.5) once the input data have been prepared off-line as

described in Chapter 3. The direct editing option is explained in Section 4.3. Certain limitations that apply to these programs are identified in Section 4.6, and the use of MIREM features is summarized in Section 4.7.

4.2 DATAIN Program Operation

This section provides instructions for operating program DATAIN. Processing during a DATAIN session is done on a series of screens. Each screen defines a single type of interaction with the user. All of the screens that can be encountered in using DATAIN are shown in Figure 20. These screens will be described in the following sections.

4.2.1 Initiating DATAIN

Procedures for accessing DATAIN will depend on the computer installation being used. Once the user has logged on and has access to the MIREM library, the DATAIN program is initiated by a command such as RUN DATAIN[1]. An access message is provided that identifies the program configuration version date.

4.2.2 DATAIN Keywords

Following the access message, the user may obtain an explanation of DATAIN keyword commands by entering HELP. These commands, which are listed in Figure 21, have the same meaning when they are entered at any point in the DATAIN session. The allowable keywords at each point are always displayed at the bottom of the screen. The HELP, CONTINUE, and QUIT commands are available at any point in the session. This feature allows the user to quickly terminate the program at any time. After the user input QUIT, a termination screen is displayed. If the user verifies the QUIT command, program DATAIN is terminated, and any unsaved data inputs are discarded.

Whenever the user is unsure about what input is being requested, HELP should be entered. A message identifying data input requirements and formats will then be provided. On the other hand, if the meaning of the data elements in the MIREM analysis is unclear, enter EXPLAIN. An explanation of the relevant MIREM terminology will be provided.

---

[1]Throughout this chapter, entries made by the user are underscored to distinguish them from messages sent by the system.

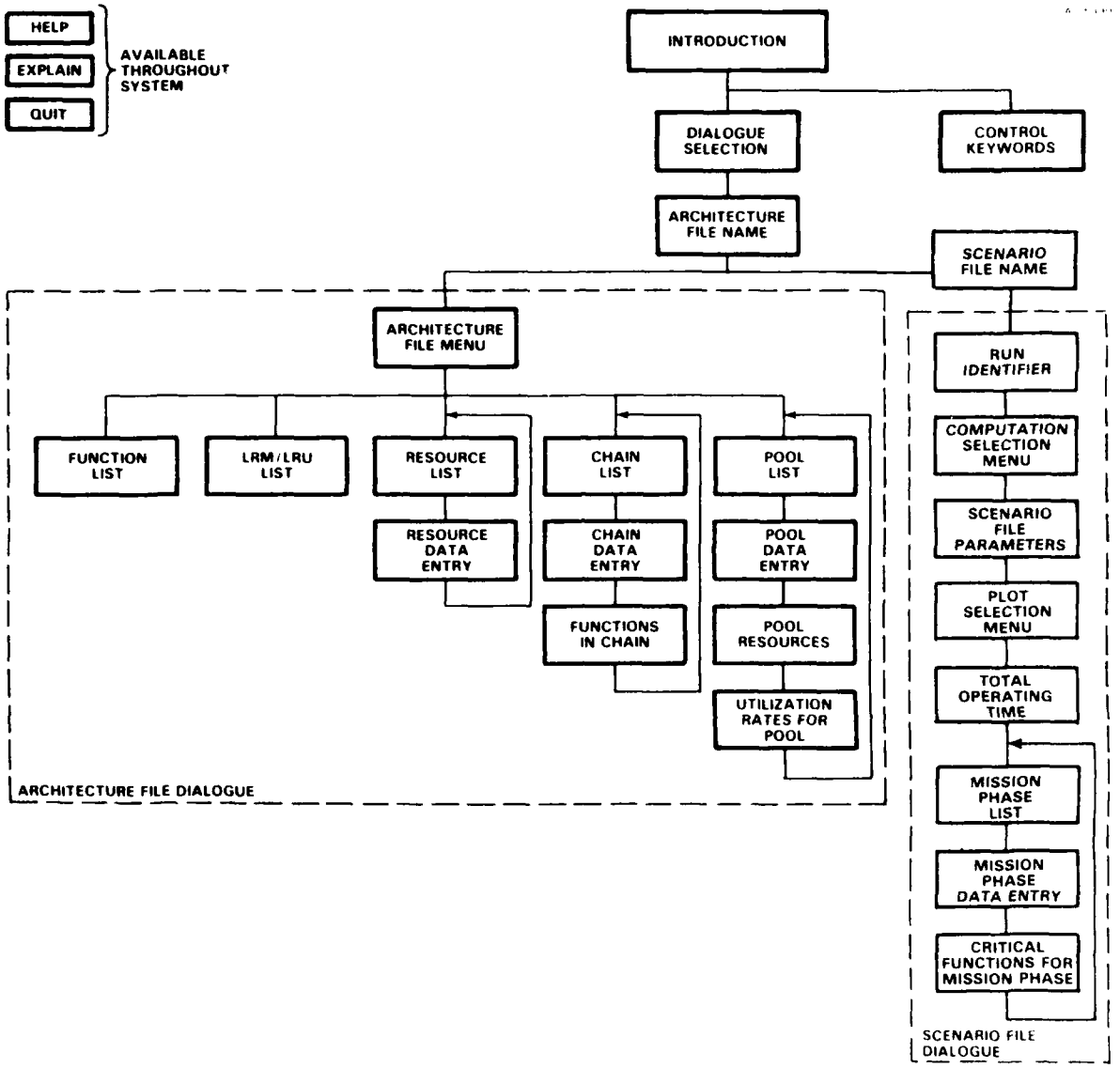


Figure 20. DATAIN Screen Flowchart.

The PAGEn command is used to view different pages of a multiscreen display. The entry PAGE2, for example, will cause the second page to be displayed.

4.2.3 File Maintenance

From the initial access or control keyword screen, the entry CONTINUE will bring the user to the dialogue selection

```
                        CONTROL KEYWORDS                    PAGE  2 OF  2

  <H>ELP      - Causes a screen to be shown which tells you how to select
                options, input data, or use the commands which are available
  <E>XPLAIN   - Causes a screen to be shown which gives additional explanation
                for the screen and how it may affect the eventual analysis
 *<P>AGEn     - Causes a multi-page screen to go to a new page where n
                should be substituted with the number of the page that you
                wish to see (e.g., 'p5' will cause page 5 to be shown)
  <C>ONTINUE  - Causes the program to accept any inputs entered and continue
                processing
  <A>DD       - Causes the program to accept the inputs entered and continue
                processing by displaying the screen(s) needed to add a new item
                to the list being processed, after the current item is done
  <B>ACK      - Causes the program to return to the previous screen in the
                dialogue sequence without storing the last changes entered
  <Q>UIT      - Causes a screen to be shown that contains termination options
                that can be selected prior to stopping the program

 *NOTE: Command is for multi-page screen only

Please enter the command:<P>AGEn, <C>ONTINUE, <Q>UIT
```

Figure 21. DATAIN Control Keywords.

menu. This screen controls the basic file maintenance activities that can be performed with DATAIN (Figure 22):

1. Create an architecture file - this file will contain all the reliability and structure data for a given design.

2. Modify an architecture file that has been created previously.

3. Create a scenario file - this file will describe a mission scenario and contain run parameters.

4. Modify a scenario file that was created previously.

Architecture and scenario file names must be valid filenames on the computer system being used. When a scenario file is selected, DATAIN checks to see if this file contains an architecture file name. If not, the user is prompted for a

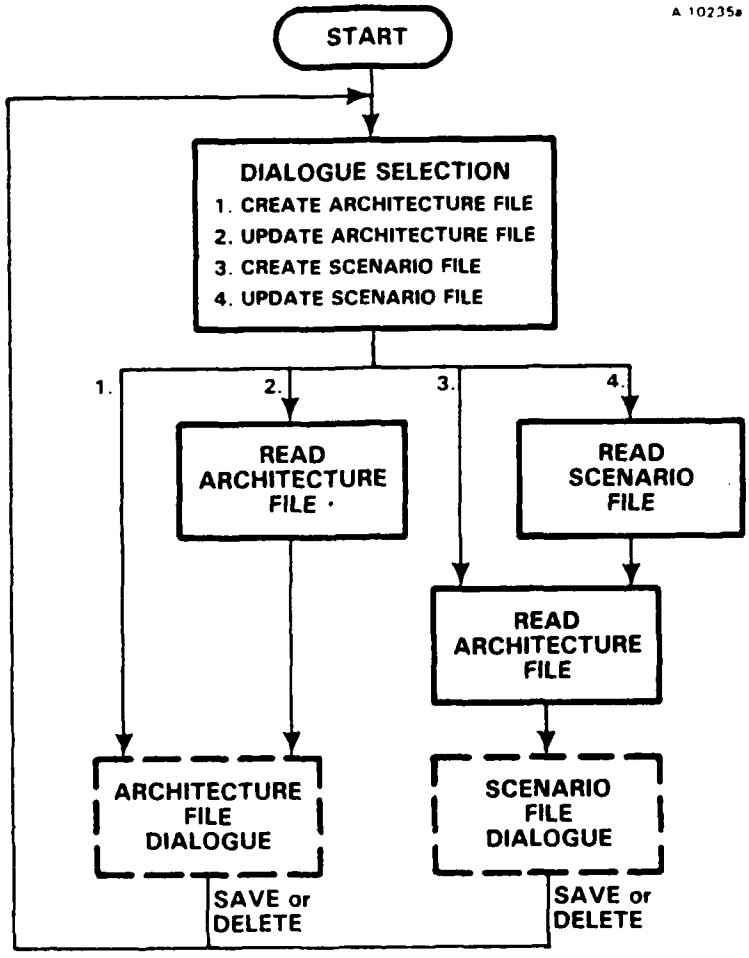


Figure 22. Overview of DATAIN Dialogue.

file name. The file name ARCHIN, for example, is specified by entering

1=ARCHIN

No spaces are allowed in this entry. This architecture file will be used as a source of functions for use in constructing the scenario file. Because of this dependence, an architecture file should be created before the associated scenario files are created. Only the function list need be entered in the architecture file. Note that a scenario file can be used with several architecture files that contain the same function list by simply changing the file name in the scenario file.

Architecture files are saved by selecting index 6 (save) in the architecture file menu screen. Scenario files are saved by entering CONTINUE from the mission phase list screen. Files

cannot be deleted or overwritten using DATAIN. The user must terminate DATAIN and use the appropriate system command in order to delete a file. Hence, to save an updated version of a file, a new file name must be assigned or, on a VAX system, a new version number may be assigned (e.g., SCENAR.DAT;2).

4.2.4 Architecture File Dialogue

Upon selecting to create or update an architecture file from the dialogue selection screen, the user enters the architecture file dialogue. This dialogue is used to enter the architecture data prepared according to Chapter 3. It consists of three general types of screens that are displayed, prompting the user for inputs:

1. Menu screen.
2. Selection list screen.
3. Data entry screen.

An example of each will be discussed.

The only menu screen is the architecture file menu (Figure 23); it is the first screen encountered in the architecture file dialogue. This screen is used to select the type of data to be entered (functions, LRMs/LRUs, resources, chains or pools). A data type is selected by entering its index number. All data types except LRMs/LRUs must be entered before an architecture file can be analyzed by MIREM. The sequence in which data types are entered should match the dependencies shown in Figure 23; e.g., functions should be entered before chains. If function data are altered after the chain data are entered, an inconsistent file might be created. The user is responsible for correcting these inconsistencies. Certain inconsistencies

```
                        ARCHITECTURE FILE MENU

Which type of data do you wish to work on?

      Index Selection Dependencies                                  Number
      ----- --------- ------------------------------------------- ------
        1.  Functions None                                          5
        2.  LRMs/LRUs None                                          5
        3.  Resources None                                          1
        4.  Chains    Functions                                     2
        5.  Pools     Functions, LRMs/LRUs, Resources, and Chains    1
        6.  Save      Not Applicable                                -

Please enter the index corresponding to your selection.
Or enter a command:<H>ELP, <E>XPLAIN, <Q>UIT
```

Figure 23. Architecture File Menu Screen.

will prevent the file from being saved. However, the user will not be alerted to many of these inconsistencies until the file is read by program MIREM. When the user is finished entering architecture data, the data are saved by entering the index 6. After a successful save, the session will return to the dialogue selection screen. The user may also cancel the architecture data that have been entered by entering QUIT.

An example of a selection list screen is the pool list (Figure 24). Selection list screens display a list of data of a certain type (e.g., pool data). When a new file is being created, no items will appear on this initial list. The user may add, repeat, change, or delete items from this list. For example, the entry

1=C 2=D ADD

will cause the data for the pool with index 1 to be displayed for the user to change. The pool with index 2 will be marked for deletion. After changes to pool 1 are completed, the user will be prompted to enter data for a new pool (add a pool). The repeat option (e.g., 1=R) allows the user to add a new pool that is identical to an existing pool. This option is particularly valuable for creating parallel chains. Type C and type S pools can often be repeated and only the chain number changed. Commands must be separated by a space or a comma. Commands of the form index=value (e.g., 1=C) may not contain spaces. To record changes and deletions in the data list and proceed to the next screen, enter CONTINUE. To cancel the changes and deletions that are displayed on the screen and return to the previous screen, enter BACK. Items that have been added to the list are not removed by the BACK command.

```
                                POOL LIST

        Pool  Chain                                No. of   Resource
 Index  No.   No.   Lrm/Lru Name     Pool Type     Branches Numbers    A/S  C/R/D
 -----  ----  ----- ---------------- ------------- -------- ---------  ---  -----
  1.     1     1    LRU1             Noncontending    1     1           A

Please enter the command: index=value, index=value, ...
    (values are 'c' for Change, 'r' for Repeat, and 'd' for Delete)
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <A>DD, <B>ACK, <Q>UIT

1=R
```

Figure 24. Pool List Screen.

One example of a data entry screen is pool data entry (Figure 25). Here the format index=value is used to enter or change parameter values. For example, the entry

3='L-BAND RF' 5=2

changes the LRM/LRU name to L-BAND RF and the number of branches to 2. Names that contain spaces must be enclosed in single quotes. Again, commands of the form index=value may not contain spaces between the index and the value. When a new pool is being created, default values will appear for some of the parameters; if these values are correct, the parameter need not be entered. To proceed to the next screen, enter CONTINUE. To cancel the changes that were made on this screen and return to the previous screen, enter BACK. In cases where several data entry screens are visited in sequence before looping back to a selection list screen (chains, pools and missions phases - see Figure 20), the BACK command cancels the changes made on all the data entry screens. Changes are recorded only when CONTINUE is typed to return to the selection list screen.

```
                              POOL DATA ENTRY

        Index Parameter                          Current Value
        ----- ------------------------------ ----------------
          1.  Pool Number                          1
          2.  Chain Number
          3.  LRM/LRU Name                       LRU1
          4.  Pool Type ('N', 'C', 'S', 'F')     N
          5.  Number of Branches                   1
          6.  Active/Standby ('A' or 'S')        A
          7.  Undetected Failure Rate            0.010
          8.  False Alarm Rate                   0.050
          9.  Min accpt level of repair            1

NOTE: (4)  'N' is Noncontending, 'C' is Contending, 'S' is Shared, and
           'F' is Chain-Fail

Please enter the command, index=value, index=value, ...
Or enter a command, <H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

2=2
```

Figure 25. Pool Data Entry Screen.

Another form of data entry screen is illustrated by the functions in chain screen (Figure 26). In this type of screen, a list of available items is displayed. A subset of these items is then specified by the user. The screen in Figure 26

```
                    FUNCTIONS IN CHAIN NUMBER 3

                          Index  Function
                          -----  --------
                           *1.   FUNC1
                           *2.   FCN2
                           *3.   FCN3
                            4.   FCN4
                            5.   FCN5

NOTE: The asterisk ('*') indicates functions selected.

Please enter the command: index to select or -index to delete.
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT
```

Figure 26. Functions in Chain Screen.

is used to specify the functions that can be processed in chain number 3. The entry

-2 4

deletes the function with index 2 from the chain and adds the function with index 4. The selected items are indicated by an asterisk before the index.

4.2.5 Scenario File Dialogue

Upon selecting to create or update a scenario file from the dialogue selection screen, the user enters the scenario file dialogue. This dialogue is used to enter:

1. A mission scenario
2. Run parameters.

The user may wish to maintain one scenario file for each mission scenario. When a particular run is needed, the run parameters (including the architecture file name) are entered into the scenario file containing the desired mission, and program MIREM is initiated.

The scenario file dialogue contains the same general types of screens as are described in Section 4.2.4. The first screen encountered is a data entry screen used to enter a run identifier. Again the format index=value is used; e.g., 1='SCENARIO FILE EDITING TEST2'. If the run identifier contains blanks,

it must be enclosed in single quotes. In the computation and plot selection menus, multiple options can be selected. The output generated by each selection is described in Chapter 5. Run parameters entered in the basic scenario file parameters screen are listed in Table 7.

The mission scenario is entered using the mission phase list screen which employs the same conventions as described in Section 4.2.4. When all mission phase data have been entered, the entry CONTINUE is used to save the data from this dialogue in the scenario file and return to the dialogue selection screen.

Table 7. Scenario File Run Parameters

Name	Description
Processing Option	'FULL' gives exact results for single-phase missions and may cause an abort for some architectures; 'QUICK' gives approximate results for parallel chains but guarantees the computation will be completed relatively quickly
Print Architecture File Report	Allows the user to verify all the model inputs that went into the run
Print Intermediate Results	Generates more detailed output as described in Chapter 5
Functions Required Simultaneously	Determines whether functions required within a phase compete for system resources. For most scenarios the appropriate value is 'YES'
Failure Rate Scale Factor	All resource failure rates are multiplied by this factor
Scheduled Maintenance	The scheduled maintenance interval (hours) used in repair analysis must be at least as large as the first total operating time
Repair Sequence	'Series': multiple repairs are performed sequentially (one maintenance crew); 'Parallel': multiple repairs are performed simultaneously (unlimited maintenance crews)

4.3 Direct Editing Option

The advanced user may elect to modify architecture and scenario files by using a system editor rather than through DATAIN. To support this direct editing option, readable formats have been adopted for these files and are defined in this section.

4.3.1 Architecture File Format

Figure 27 shows the format of the architecture file. The architecture file generated by program DATAIN is shown in Figure 28. The file must contain at least one card of each type except LRU, GROUP, GROUP RESOURCE, GROUP UTILIZATION, and comments. The order restrictions are:

1. Chain function cards must be immediately after the chain card to which they refer.

2. Pool resource and pool utilization cards must be immediately after the pool card to which they refer, in that order.

A 36409

FUNCTION	fcn-name-list
LRU	lru-name-list
RESOURCE	r-number, r-qty, f.r., r/i, mttr, r-name
CHAIN	p-chn-number, { 0 / s-chn-number }, c-name
CHAIN FUNCTION	{ p-chn-number / s-chn-number }, fcn-list
POOL	p-number, { *p-chn-number* / s-chn-number }, { 0 / lru-index }, p-type, no-branches, a/s, undetected-f.r., false-alarms, repair-level
POOL RESOURCE	p-number, r-list
POOL UTILIZATION	p-number, u-list
*	comment
GROUP	g-number, p-chn-number, { 0 / lru-index }, g-type, no-subgr
GROUP RESOURCE	g-number, subgr-list
GROUP UTILIZATION	g-number, u-list

Figure 27. Architecture File Format.

```
*************************************************************************
* FUNCTION CARD: LIST OF FUNCTION NAMES                                 *
*************************************************************************
FUNCTION  'GPS      ' 'UHF      ' 'SINC     '
*************************************************************************
* LRM/LRU CARD: LIST OF LRM/LRU NAMES                                   *
*************************************************************************
LRU       'FRONTEND         ' 'DIGITALA         ' 'DIGITALB         '
*************************************************************************
* RESOURCE CARD: RESOURCE NO., QUANTITY, FAILURE RATE, TYPE, MTTR, NAME*
* NOTE: FAILURE RATE IS IN PER MILLION HOURS                            *
*************************************************************************
RESOURCE        1       1      10 R 4.0 L-BAND ANTENNA CONNECTOR
RESOURCE        2       2      15 R 3.5 L-BAND RECEIVER
RESOURCE        3       2       5 R 2.0 2 X 3 L-BAND SWITCH PORTS
RESOURCE        4       1      10 R 2.5 LOW-BAND ANTENNA SWITCH
RESOURCE        5       2      95 R 4.0 LOW-BAND RECEIVER
RESOURCE        6       2       5 R 3.0 2 X 5 LOW-BAND SWITCH PORTS
RESOURCE        7       5     300 R 2.0 PREPROCESSOR
RESOURCE        8       2     100 R 5.0 SIGNAL PROCESSOR
RESOURCE        9       2      20 R 4.5 POWER SUPPLY
RESOURCE       10       2      20 R 4.0 SDU I/O
RESOURCE       11       2     100 R 6.0 CONTROLLER
*************************************************************************
* CHAIN CARD: PRIMARY CHAIN NUMBER, PARALLEL CHAIN NUMBER, CHAIN NAME   *
* NOTE: PARALLEL CHAIN NUMBER IS 0 WHEN CHAIN IS SERIES CHAIN           *
*************************************************************************
* CHAIN FUNCTION CARD: CHAIN NUMBER, LIST OF FUNCTION INDICES           *
*************************************************************************
CHAIN              1    0 FRONT END
CHAIN FUNCTION     1     1  2  3
CHAIN              2    3 DIGITAL
CHAIN FUNCTION     2     1  2  3
CHAIN FUNCTION     3     2  3
*************************************************************************
* POOL CARD: POOL NUMBER, CHAIN NO, LRU INDEX, POOL TYPE, BRANCHES,     *
* ACTIVE/STANDBY, UNDETECTED FAILURES, FALSE ALARMS, MIN REPAIR LEVEL   *
* NOTE: FOR POOL TYPE, 'N' IS NONCONTENDING, 'C' IS CONTENDING,         *
*                  'S' IS SHARED, AND 'F' IS CHAIN-FAIL                 *
*************************************************************************
* POOL RESOURCE CARD: POOL NUMBER, LIST OF RESOURCE NUMBERS             *
*************************************************************************
* POOL UTILIZATION CARD: POOL NUMBER, LIST OF FUNCTION UTILIZATIONS     *
*************************************************************************
POOL         10    1    1    N    1 A  0.1   0.05  1
POOL RESO    10    1   2   2   3   3
POOL UTIL    10  1.00  0.00  0.00
POOL         11    1    1    N    1 A  0.005 0.001 1
POOL RESO    11    4
POOL UTIL    11  0.00  1.00  1.00
POOL         12    1    1    N    2 S  0.02  0.01  1
POOL RESO    12    5   6
POOL UTIL    12  0.00  1.00  1.00
POOL         13    2    2    C    3 A  0.01  0.01  2
POOL RESO    13    7
```

Figure 28. Architecture File for the Example Architecture.

```
POOL UTIL    13  2.00  1.00  1.00
POOL         14    2     2    S     1 A  0.02  0.005 1
POOL RESO    14    8
POOL UTIL    14  0.80  0.10  0.40
POOL         15    2     2    F     1 A  0.    0.01  1
POOL RESO    15    9
POOL UTIL    15  1.00  1.00  1.00
POOL         16    2     2    N     1 A  0.01  0.005 1
POOL RESO    16   10
POOL UTIL    16  0.00  0.00  1.00
POOL         17    2     2    N     1 A  0.05  0.02  1
POOL RESO    17   11
POOL UTIL    17  1.00  1.00  1.00
POOL         13    3     3    C     2 A  0.01  0.01  2
POOL RESO    13    7
POOL UTIL    13  2.00  1.00  1.00
POOL         14    3     3    S     1 A  0.02  0.005 1
POOL RESO    14    8
POOL UTIL    14  0.80  0.10  0.40
POOL         15    3     3    F     1 A  0.    0.01  1
POOL RESO    15    9
POOL UTIL    15  1.00  1.00  1.00
POOL         16    3     3    N     1 A  0.01  0.005 1
POOL RESO    16   10
POOL UTIL    16  0.00  0.00  1.00
POOL         17    3     3    N     1 A  0.05  0.02  1
POOL RESO    17   11
POOL UTIL    17  1.00  1.00  1.00
```

Figure 28. (Concluded).

3. Group resource and group utilization cards must be immediately after the group card to which they refer, in that order.

4. Groups must be ordered so that all pools and groups in the subgroup list occur before that group.

5. All function cards should precede all chain cards.

6. All function, LRU, resource, and chain cards should precede all pool cards.

These cards contain the following elements:

1. fcn-name-list - a list of 8-character function names. Names containing blanks should be enclosed in single quotes. Names should be separated by blanks or commas. The order of a function in the list determines its index; e.g., the first function has a index of 1.

2. lru-name-list - a list of 16-character LRU names. See fcn-name-list.

3. r-number - resource type number; a unique integer from 1 to 999.

4. r-qty - the number of resources of a given type in the system; an integer from 1 to 99.

5. f.r. - failure rate in failures per million hours; an integer from 1 to 9999.

6. r/i - each resource type is a RESOURCE or INTERCONNECTION.

7. mttr - mean time to repair this resource type, in hours.

8. r-name - a 30-character resource name; not enclosed in single quotes.

9. p-chn-number - primary chain number; an integer from 1 to 99 (must be unique on chain cards).

10. s-chn-number - secondary chain number; an integer from 1 to 99 (must be unique and distinct from primary chain numbers on chain cards).

11. c-name - a 30-character chain pair name; not enclosed in single quotes.

12. fcn-list - a list of function indexes; each index is an integer from 1 to the number of functions listed on function cards.

13. p-number - pool number; an integer from 1 to 999, unique except that two occurrences are allowed if the chain numbers listed on their pool cards form a parallel chain pair.

14. lru-index - LRU index; an integer from 1 to the number of LRUs listed on LRU cards.

15. p-type - pool type; N, C, S, or F.

16. no.-branches - number of branches; an integer from 1 to 99.

17. a/s - each pool uses ACTIVE or STANDBY redundancy.

18. undetected-f.r. - the fraction of failures in a pool that are undetected; from 0 to 1.

19. false-alarms - the ratio of false alarms to failures in a pool; nonnegative.

20. repair-level - the minimum number of branches that must be functioning in a pool for the system to be allowed to perform another mission without maintenance; an integer from 0 to no.-branches.

21. r-list - list of r-numbers; separated by blanks or commas.

22. u-list - list of utilization rates. Sequence in the list identifies the function index to which the rate applies. Length of the list should equal the number of functions listed on function cards. Each rate is a real number between 0 and no.-branches (for pools) or the length of the subgr-list (for groups).

23. g-number - group number; a unique integer from 1000 to 1999.

24. g-type - group type; must be 'N.'

25. no.-subgr - number of subgroups; not used, set to 1.

26. subgr-list - list of r-numbers and g-numbers; separated by blanks or commas, no repetition allowed.

4.3.2 Scenario File Format

Figure 29 shows the format of the scenario file. A scenario file generated by program DATAIN is shown in Figure 30. The file should contain at least one card of each type except phase, phase function, and comment cards. Phase function cards must be immediately after the phase card to which they refer. The sequence of the phase cards determines the order of phases in the mission. These cards contain the following elements:

1. runid - a 72-character name identifying the run.

2. filename - architecture file name; a legal system file name.

3. s-factor - failure rate scale factor; a positive real number.

4. total-operating-time-list - a list of positive real numbers separated by blanks, in units of hours.

A 36408

RUNID	runid
HARDWARE	filename
COMPUTE	[MCSP] [PHASE-BY-PHASE] [MTBCF] [MTBFF] [LRU] [REPAIR] [BIT] [FULLBIT]
PLOT	[MCSP] [PHASE-BY-PHASE] [MTBCF] [MTBFF] [LRU] [REPAIR]
QUICK	YES / NO
PRINT	HARDWARE / INTERMEDIATE ; YES / NO
SIMULTANEOUS	YES / NO
SCALE	s-factor
TIME	total-operating-time-list
TMAINTENANCE	scheduled-maintenance-interval
REPSEQUENCE	SERIES / PARALLEL
PHASE	ph-index, ph-duration, ph-name
PHASE FUNCTION	ph-index, fcn-list
*	comment

Figure 29. Scenario File Format.

```
RUNID TEST1
*************************************************************************
COMPUTE              MCSP PHASE LRU MTBCF MTBFF REPAIR BIT FULLBIT
PLOT                 MCSP  MTBCF PHASE REPAIR
QUICK                 YES
PRINT HARDWARE        YES
PRINT INTERMEDIATE     NO
SIMULTANEOUS          YES
SCALE                      1.0
TIME                       3.0 HOURS
TMAINTENANCE             100. HOURS
REPSEQUENCE          PARALLEL
*************************************************************************
* PHASE CARD: PHASE INDEX, TIME-OF-PHASE (HOURS), PHASE NAME           *
*************************************************************************
* PHASE FUNCTION CARD: PHASE INDEX, LIST OF FUNCTION INDICES           *
*************************************************************************
PHASE                1      1.50 PHASE 1
PHASE FUNCTION       1      2
PHASE                2      1.00 PHASE 2
PHASE FUNCTION       2      1 3
PHASE                3      0.50 PHASE 3
PHASE FUNCTION       3      2
```

Figure 30. Sample Scenario File.

5. scheduled-maintenance-interval - real number greater than or equal to the first total-operating-time, in units of hours.

6. ph-index - phase index; an integer from 1 to the number of phases, equal to the sequence of this phase card among all the phase cards.

7. ph-duration - phase duration; a positive real number, in units of hours.

8. phase-name - a 30-character phase name.

9. fcn-list - list of function indices, separated by blanks or commas; integers from 1 to the number of functions, unique within a phase.

4.4 MIREM Program Operation

Once architecture and scenario files have been created using program DATAIN, program MIREM is used to compute model results and to generate reports desplaying the data in these files. Procedures for accessing MIREM will depend on the computer installation being used. For a VAX 11/780 system, the MIREM program may be run by entering:

```
@MIREM scenario-filename
```

where scenario-filename is the system name of a valid scenario file. Note that the scenario-filename may be left out, in which case the interactive user will be prompted for the scenario-filename.

Generally, the scenario file contains a hardware card with the hardware file name. In this case the user does not need to worry about the hardware file. However, if the user uses the direct editing option (Section 4.3) and creates a scenario file without a hardware card, then the user should enter:

```
@MIREM scenario-filename    hardware-filename
```

where hardware-filename is the system name of a valid hardware file.

The results are written to a file which by default is named MIREM.OUT on a VAX system. This file may be examined on-line. On most systems it will be printed automatically. The output may be written to a different file by entering:

@MIREM scenario-filename hardware-filename output-filename plot-filename

where output-filename specifies a file to which to write output reports and plot-filename specifies a file to which to write the plot data.

4.5 MPLOT Program Operation

When program MIREM has been run on a scenario file containing a plot card, and a plot file has been generated, program MPLOT is used to generate plots. Use of MPLOT requires linking with the DI3000 graphics package and associated device drivers. The MPLOT program is run, on a VAX 11/780 system, by entering

RUN MPLOT

The user will then be prompted for the plot file name and the device number. For most users, the device number will be 1. Procedures for directing the plot output to a graphics printer will depend on the computer installation being used.

4.6 Limitations

Certain limits have been set on the size of model that can be analyzed by MIREM. These limits are shown in Table 8. They result in virtual memory requirements of 205,000 bytes for program MIREM, 275,000 bytes for program DATAIN, and 60,000 bytes for program MPLOT on a VAX 11/780. Also, a disk space of 3.6 megabytes should be allocated for the combined software.

Three additional restrictions have been imposed to prevent extremely long run times. The worst-case run time grows exponentially with the number of functions listed within a mission that appear in the function list for both chains in a parallel chain pair. The number of functions in this category is limited to eight.

When the full processing option is selected, MIREM iterates over "substantially different" function allocations to parallel chains (see Appendix A). The number of allocations is limited to 15.

Table 8. MIREM Size Limitations

Category	Maximum Number
Functions	40
LRMs/LRUs	200
Resources	200
Chain pairs	10
Pools and groups	200
Mission phases	10
Branches per pool	30
Resources per branch	10
Subgroups per group	10
Functions per phase	40
	Maximum Length (Characters)
Function name	8
LRM/LRU name	16
Resource name	30
Chain pair name	30
Mission phase name	30
Run identifier name	72

The run time for MTBCF calculations is roughly proportional to the number of integration time steps. For typical inputs, the integration routine stops after 20 to 30 steps. However, it is conceivable that for some input data the routine would continue for many more steps. The number of steps is limited to 50. If this limit is reached, a warning message will be printed, and the accuracy of the MTBCF result is suspect.

4.7 Compatibility of Features

Not all of the features of this version of MIREM can be used at the same time. The computational approaches used to obtain some of the results in this version preclude the consideration of some of MIREM's other features. These limitations are summarized in Table 9. In some cases, a computation cannot be performed at all; MIREM will abort the run or, at best, skip the computation. These combinations (X) should be

Table 9. Compatibility of Features

Output Option	All Outputs				Bit, Fullbit	Repair			MCSP, MTBCF	Phase-by-Phase	MTBFF	LRU/LRM Budget
Feature A / Feature B	Full Processing Option	Standby Redundancy	Groups	Parallel Chains	Imperfect Switching	Scheduled Maintenance	Repair at Degraded Level	MTTR				
Multiple Phases	X	N										
Full Processing Option					N							
Standby Redundancy			X[a]		N			N				
Groups				X[b]	X		X					
Parallel Chains							N[c]					
Imperfect Switching						N	N	N	N	N	N	N
Scheduled Maintenance							N		N		N	N
Repair at Degraded Level									N		N	N

[a] Standby redundancy is not allowed within chains containing groups.

[b] Groups are not allowed within parallel chains.

[c] Reliability is not computed. Availability is computed, but parallel chain redundancy is not considered in the repair decision.

KEY

X: Incompatible; MIREM will abort or skip computation

N: Neglected; MIREM will neglect feature B when Feature A is present or when computing this output option

avoided. In other cases, MIREM neglects a feature (feature B) that would affect the result. The user should be aware that some of the input data is thus being neglected and should exercise caution in comparing reliability results from different output options, since different features may have been neglected.

5. SAMPLE STUDIES

5.1 Mission Effectiveness Analysis

MIREM can be used to evaluate reliability against a specific mission scenario by using the MCSP, phase-by-phase, or BIT output options. When conducting any analysis, it is advisable to obtain architecture and scenario file reports so that all of the model inputs are known. Figure 31 shows the architecture file report for the example architecture introduced in Chapter 3; portions of the report shown in Chapter 3 are not repeated here. Figure 32 shows a scenario file report. The mission consists of three phases, with a total duration of 3.0 hours. During phases one and three, only UHF (function 2) is required. During phase two, GPS and SINCGARS are required simultaneously.

The MCSP and pool budget results are shown in Figure 33. Contributions to MCSP are broken down by chain pair and by pool. For parallel chain pairs, the manner in which MCSP is computed does not allow visibility into individual pools; instead, the MCSP for each type of pool is given. It may be possible to perform all the functions on one parallel chain even if the other chain is down (a type F pool failure). The additional MCSP achieved because of this capability is listed for the primary and secondary chain in the pair. In this example, neither chain can support the entire mission, and the additional MCSP is zero. The immediate repair MTBCF of 2500 hours is calculated from MCSP, assuming that the system is fully repaired after each mission (mission length is taken from the total operating time input). This number will always be larger than the deferred repair MTBCF described in Section 5.2.

The equations used for the MCSP and pool budget output contain the approximation that each individual phase must be performed at the end of the mission rather than in sequence. This "phase approximation" generates pessimistic MCSP results if the final mission phases are less demanding than previous phases. It should be noted that logistics/sustainability measures, such as MTBCF, are not affected by this approximation because they address end-of-mission or multiple-mission status. An alternative computational approach that avoids this approximation is used for the phase-by-phase MCSP output.

ARCHITECTURE FILE REPORT

POOL REPORT

CHAIN NUMBER	POOL NUMBER	LRM/LRU NAME	NUMBER BRANCHES	POOL FAILURE RATE •	POOL TYPE	REDUN-DANCY	MINIMUM LEVEL REPAIR	UNDETECTED FAILURE RATE	FALSE ALARM RATE	RESOURCE NUMBER	RESOURCE FAILURE RATE •	RESOURCE NAME
1	10	FRONTEND	1	50	NONCONTENDING	ACTIVE	1	0 100	0 050	1	10	L-BAND ANTENNA C
										2	15	L-BAND RECEIVER
										2	15	L-BAND RECEIVER
										3	5	2 X 3 L-BAND SWI
										3	5	2 X 3 L-BAND SWI
	11	FRONTEND	1	10	NONCONTENDING	ACTIVE	1	0 005	0.001	4	10	LOW-BAND ANTENNA
	12	FRONTEND	2	200	NONCONTENDING	STANDBY	1	0 020	0 010	5	95	LOW-BAND RECEIVE
										6	5	2 X 5 LOW-BAND S
2	13	DIGITALA	3	900	CONTENDING	ACTIVE	2	0.010	0 010	7	300	PREPROCESSOR
	14	DIGITALA	1	100	SHARED	ACTIVE	1	0 020	0 005	8	100	SIGNAL PROCESSOR
	15	DIGITALA	1	20	CHAIN-FAIL	ACTIVE	1	0 000	0 010	9	20	POWER SUPPLY
	16	DIGITALA	1	20	NONCONTENDING	ACTIVE	1	0 010	0 005	10	20	SDU I/O
	17	DIGITALA	1	100	NONCONTENDING	ACTIVE	1	0 050	0 020	11	100	CONTROLLER
3	13	DIGITALB	2	600	CONTENDING	ACTIVE	2	0 010	0 010	7	300	PREPROCESSOR
	14	DIGITALB	1	100	SHARED	ACTIVE	1	0 020	0 005	8	100	SIGNAL PROCESSOR
	15	DIGITALB	1	20	CHAIN-FAIL	ACTIVE	1	0 000	0 010	9	20	POWER SUPPLY
	16	DIGITALB	1	20	NONCONTENDING	ACTIVE	1	0 010	0 005	10	20	SDU I/O
	17	DIGITALB	1	100	NONCONTENDING	ACTIVE	1	0 050	0 020	11	100	CONTROLLER
SYSTEM FAILURE RATE				2240								

(•) FAILURE RATE IN PER MILLION HOURS

ARCHITECTURE FILE REPORT

FUNCTION UTILIZATION BY POOL

CHAIN NUMBER	POOL NUMBER	FUNCTION GPS	UHF	SINC
1	10	1.00	0.00	0.00
	11	0.00	1.00	1.00
	12	0.00	1.00	1.00
2	13	2.00	1.00	1.00
	14	0.80	0.10	0.40
	15	1.00	1.00	1.00
	16	0.00	0.00	1.00
	17	1.00	1.00	1.00
3	13	2.00	1.00	1.00
	14	0.80	0.10	0.40
	15	1.00	1.00	1 00
	16	0.00	0.00	1.00
	17	1.00	1.00	1.00

Figure 31. Architecture File Report for the Example Architecture.

This output gives an upper and lower bound for the probability of successfully completing each mission phase, culminating in MCSP for the entire mission (Figure 34). For this example, phase three is less demanding than phase two, and the phase approximation MCSP (0.9988) is actually below the lower bound MCSP (0.9989). The true MCSP lies between the lower and upper

SCENARIO FILE REPORT

COMPUTATION/PLOT SELECTIONS:

1. MCSP AND POOL/CHAIN BUDGET; IMMEDIATE REPAIR MTBCF (PLOT)
2. PHASE-BY-PHASE MCSP (PLOT)
3. DEFERRED REPAIR MTBCF (PLOT)
4. MTBFF - MEAN TIME BETWEEN FUNCTION FAILURES (NO PLOT)
5. LRM/LRU BUDGET (NO PLOT)
6. REPAIR POLICY (PLOT)
7. TESTABILITY FACTORS - BIT OPTION (NO PLOT)
8. TESTABILITY FACTORS - BIT MTBCF OPTION (NO PLOT)

NOTES:
MCSP - MISSION COMPLETION SUCCESS PROBABILITY
MTBCF - MEAN TIME BETWEEN CRITICAL FAILURES

BASIC SCENARIO FILE PARAMETERS:

1.	PROCESSING OPTIONS:	QUICK
2.	PRINT HARDWARE FILE REPORT?:	YES
3.	PRINT INTERMEDIATE RESULTS?:	NO
4.	FUNCTIONS REQUIRED SIMULTANEOUSLY?:	YES
5.	FAILURE RATE SCALE FACTOR:	1.0
6.	TOTAL OPERATING TIME (HOURS):	3.00
7.	REPAIR SEQUENCE (MULTIPLES):	PARALLEL
8.	SCHEDULED MAINTENANCE INTERVAL:	100.0

MISSION PHASE LIST

INDEX	PHASE NAME	LENGTH (HOURS)	CRITICAL FUNCTIONS
1.	PHASE 1	1.50	2
2.	PHASE 2	1.00	1,3
3.	PHASE 3	0.50	2

Figure 32. Scenario File Report.

bounds. In fact, the lower bound is very close to the true MCSP for this mission.

The lower bound will generally be "tight" if the mission has only one demanding phase or the last demanding phase includes the other phases' function requirements; the upper bound will be tight if the system does not contain much fault tolerance (failure resiliency near 1.0) or if only the first phase is demanding. Unfortunately, for many missions, particularly those with many phases, the bounds diverge widely. Therefore, it is recommended that the phase-by-phase output be used to compare mission phases and assess weapons delivery or other success probabilities within a mission, and that the MCSP and pool budget output be used to compare missions as a whole.

MCSP AND BUDGET OUTPUT OPTION

CHAIN NUMBER	CHAIN NAME	SERIES/PARALLEL
1	FRONT END	SERIES
	POOL NUMBER	POOL MCSP
	10	0.999850
	11	0.999970
	12	1.000000
	CHAIN MCSP:	0.999820
2,3	DIGITAL	PARALLEL
	TYPE N POOL MCSP:	0.999700
	TYPE C POOL MCSP:	1.000000
	TYPE S POOL MCSP:	0.999400
	TYPE F POOL MCSP:	0.999880
	CHAIN MCSP (BOTH CHAINS OPERABLE):	0.998980
	CHAIN MCSP (PRIMARY CHAIN ONLY):	0.000000
	CHAIN MCSP (SECONDARY CHAIN ONLY):	0.000000
	CHAIN MCSP:	0.998980
IMMEDIATE REPAIR MTBCF:		2500.
TOTAL SYSTEM MCSP AT TIME	3.00 HOURS:	0.998801

Figure 33. MCSP and Pool Budget Output.

PHASE-BY-PHASE MCSP REPORT

TIME INTO MISSION	CURRENT PHASE	SUCCESS PROBABILITY LOWER BOUND	UPPER BOUND
0.00	START OF MISSION	1.000000	1.000000
1.50	PHASE 1	0.999985	0.999985
2.50	PHASE 2	0.998985	0.999585
3.00	PHASE 3	0.998955	0.999580
SYSTEM MCSP		0.998955	0.999580
IMMEDIATE REPAIR MTBCF(HOURS)		2870	7140

Figure 34. Phase-by-Phase MCSP Output.

The BIT option can be used to investigate the impact of imperfect switching, due to incomplete testability, on MCSP. Figure 35 gives upper and lower bounds on MCSP, taking into account imperfect BIT. Testability causes seven or eight failures per 100,000 missions for this example. For systems with high undetected failure rates or high failure resiliency, the contribution of BIT will be more significant. The probability of a false mission failure indication is shown to be less than two in 100,000.

TESTABILITY FACTORS REPORT
BIT OPTION

MISSION DURATION = 3.00 HOURS

	LOWER BOUND	UPPER BOUND
PERFECT BIT MCSP	0.99880065	0.99880065
IMPERFECT BIT MCSP	0.99871965	0.99873525
PROBABILITY OF MISSION FAILURE DUE TO BIT	0.00008100	0.00006540
MISSION FAILURE FALSE ALARM PROBABILITY	0.00000227	0.00001794

Figure 35. Testability Factors (BIT) Output.

5.2 Logistics/Sustainability Analysis

MIREM can be used to evaluate logistics impacts in terms of high removal rate items and bare-base[2] sustainability by using the LRM/LRU budget, MTBCF, and full BIT output options. The LRM/LRU budget results for the input files shown in Section 5.1 are listed in Figure 36. It should be noted that the relative contribution of LRMs/LRUs depends on the operating

LRM/LRU BUDGET REPORT

TOTAL OPERATING TIME: 46.00 HOURS
MISSION DURATION: 3.00 HOURS

LRM/LRU	MARGINAL MCSP (CUMULATIVE)	RELATIVE CONTRIBUTION (CUMULATIVE)	PROBABILITY OF REMOVAL UPON REPAIR
FRONTEND	0.845040	0.130	0.103612
DIGITALA	0.921967	0.562	0.588351
DIGITALB	0.880019	0.326	0.367499
INTERCONNECTIONS	N/A	N/A	N/A

SYSTEM MCSP (CUMULATIVE): 0.821932
SYSTEM MCSP (LAST MISSION): 0.998496

Figure 36. LRM/LRU Budget Output.

[2]Bare-base maintenance scenarios are characterized by no off-equipment LRM/LRU repair and limited spares with which to perform on-equipment repair.

time since repair. The total operating time has been set to MTBF to give a better indication of high removal rate items under a deferred repair policy. The value MTBCF is also useful. This system contains three LRUs; no interconnections were modeled. The Digital A LRU is the largest contributor to critical failures (56%). If this LRU never failed, cumulative MCSP would increase from 0.82 to 0.92. The probability of removing the LRU upon repair is about 0.59; summing these numbers across LRUs shows that the average number of LRUs removed is 1.06. These probabilities can be combined with MTBCF to give the LRU removal rates under deferred repair, which will be somewhat lower than the traditional LRU failure rate.

The MTBCF results for the same architecture and mission are shown in Figure 37. The MTBCF of 1459 hours is based on the assumption that no repairs are made before critical failure. Bare-base operations, for example, generally have this characteristic. In contrast, the immediate repair MTBCF of 2499 hours assumes that the system is fully repaired after each 3-hour mission. For highly fault-tolerant systems,

MTBCF REPORT

MEAN TIME BETWEEN CRITICAL FAILURES (MTBCF):	1459. HOURS
MEAN TIME BETWEEN FAILURES (MTBF):	446. HOURS
FAILURE RESILIENCY:	3.27

INTEGRATION INTERVALS

MIDPOINT (HOURS)	WIDTH (HOURS)	FAILURE RATE (E-6/HOURS)	AREA (HOURS)
5.58	11.16	400.1	11.14
33.48	44.64	401.5	44.05
145.09	178.57	417.7	168.34
591.52	714.29	553.4	542.43
1143.58	389.84	733.5	208.65
1502.95	328.91	830.4	132.99
1744.00	153.20	885.9	50.14
1944.09	246.97	926.2	67.65
2182.62	230.09	969.1	50.27
2402.97	210.62	1004.2	37.01
2607.09	197.61	1033.2	28.19
2800.19	188.58	1058.1	21.98
2985.36	181.76	1079.8	17.38
3173.08	193.70	1100.0	15.11
3386.50	233.13	1120.8	14.38
3647.67	289.22	1143.8	13.34
3981.85	379.13	1169.5	12.00
4441.52	540.20	1199.5	10.18
5153.89	884.55	1236.4	7.64
6532.02	1871.70	1285.3	4.43
10758.80	6581.86	1350.4	1.29

NUMBER OF MCSP EVALUATIONS:	24
NUMBER OF INTERVALS:	21
STOPPING POINT:	14049.73 HOURS
FINAL MCSP:	0.000000
CONSTANT FAILURE RATE MTBCF:	2499. HOURS

Figure 37. MTBCF Output.

the immediate repair MTBCF result can be unrealistically high because it does not take into account undetected and intermittent failures, replacement of the wrong item, and other factors that degrade real-world maintenance performance. The degree of fault tolerance is best seen in the failure resiliency (3.27), which roughly corresponds to the number of resource failures until critical failure.

The additional output in Figure 37 (starting with "integration intervals") was generated by requesting intermediate results to be printed. This output traces the progress of the numerical integration algorithm that computes MTBCF. The algorithm required 24 MCSP evaluations and therefore used nearly 24 times as much computer time as the MCSP and pool budget output. For this reason, MTBCF outputs are not recommended for test or sensitivity runs. The algorithm stopped at an operating time of 14,000 hours, at which point the probability of no critical failures (MCSP) was insignificant. For constant failure rate systems (little fault tolerance), the algorithm will stop much sooner and still give accurate results (see Appendix A.6). Plotting the failure rate against the operating time since repair (use the "midpoint" column in Figure 37) illustrates the impact on MCSP of performing missions with degraded systems under a deferred repair policy. Figure 38 shows that the failure rate (and hence, the mission failure probability) doubles for a system that has operated for the MTBCF without a critical failure and without repair.

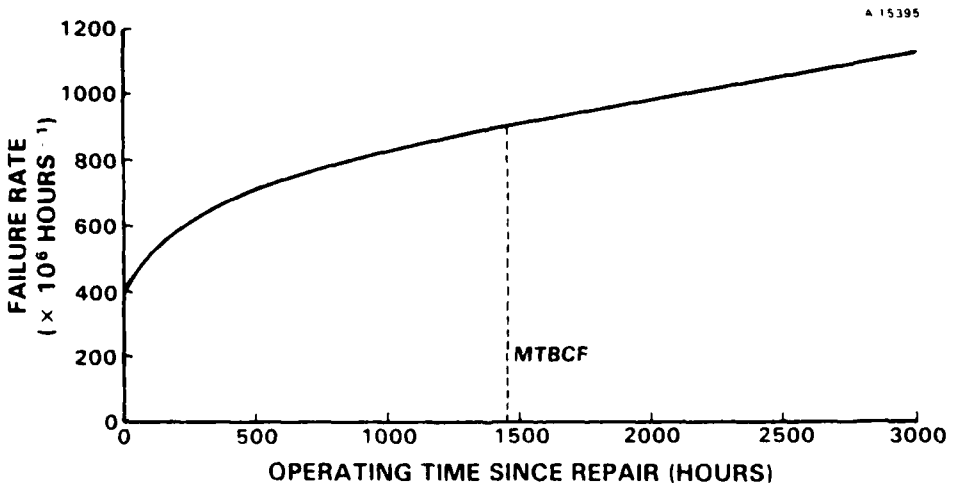


Figure 38. Failure Rate as a Function of Operating Time.

MTBCF results that include testability factors can be obtained by selecting the full BIT output (Figure 39). For this example, imperfect BIT reduces MTBCF from 1459 hours to between 1427 and 1436 hours. Again, systems with high undetected failure rates or high failure resiliency will have a more significant degradation.

TESTABILITY FACTORS REPORT
BIT MTBCF OPTION

MISSION DURATION = 3.00 HOURS

	LOWER BOUND	UPPER BOUND
MTBCF	1427.09	1436.09
MTBF	446.43	446.43
FAILURE RESILIENCY	3.20	3.22
MCSP AT TIME T	0.998719647	0.998735245

Figure 39. Testability Factors (BIT) MTBCF Output.

All of the performance measures described up to this point depend on the mission scenario. A useful measure that is not related to a mission is MTBFF for each function performed by the system. Figure 40 lists the success probability (MCSP), MTBFF, and failure resiliency for each function. GPS has a much lower failure resiliency than the other functions because the L-Band Front End and several Digital A resources are single-point critical failures with respect to GPS.

MTBFF REPORT

TOTAL OPERATING TIME: 3.00 HOURS

FUNCTION	MCSP	MTBCF IMMEDIATE REPAIR	MTBCF DEFERRED REPAIR	FAILURE RESILIENCY
GPS	0.999488	5853.	1967.	4.41
UHF	0.999970	99007.	4224.	9.46
SINC	0.999970	98855.	4055.	9.08

Figure 40. MTBFF Output.

5.3 Repair Policy Analysis

MIREM can be used to evaluate the impact of innovative repair policies on mission reliability and availability, using the repair output option. The results for the example of Section 5.1 are shown in Figure 41. Average MCSP, or equivalently, MTBCF, is highest for the immediate repair policy and lowest for the deferred repair policy, which maintains the system in the poorest state of repair. Reliability measures cannot be computed for the repair at degraded level policy because this example contains parallel chains. Conversely, MTBMA is least for immediate repair. The scheduled maintenance policy, with a maintenance interval of 100 hours, only increases MTBMA from 446 to 447 hours. By repairing at degraded levels, an average of almost two faults are repaired at each maintenance time, extending MTBMA to 745 hours. Under deferred repair MTBMA is merely MTBCF - maintenance corresponds to critical failures.

Because the model neglects the reduction in failure rate that results from operating with failed resources, the same number of repairs must be performed under each maintenance policy. Availability is greater for policies with large MTBMA, because multiple repairs are performed simultaneously. If series (sequential) repair were selected, all policies would give the same availability.

REPAIR POLICY REPORT

MISSION DURATION = 3.00 HOURS

QUANTITY	IMMEDIATE REPAIR	DEFERRED REPAIR	SCHEDULED MAINTENANCE	REPAIR AT DEGRADED LEVEL
AVERAGE MCSP	0.998800646	0.997945310	0.998791206	** N/A **
MTBCF	2500.	1459.	2480.	** N/A **
MTBMA	446.43	1458.57	488.42	745.35
MTTR	2.24	4.00	2.27	2.42
INHERENT AVAILABILITY	0.99501	0.99727	0.99537	0.99676

Figure 41. Repair Output Option.

5.4 Sensitivity Analysis

MIREM can be used to quickly evaluate the impact of system changes on reliability, and can thus be a useful aid for designing fault tolerance into complex systems. The sensitivity to resource reliability, redundancy, and reconfigurability can be easily analyzed. The MCSP and pool budget output option is recommended for sensitivity analysis. If the mission length is used as the total operating time, then the study will show the impacts on mission reliability for a full-up system. If the system MTBCF (or MTBF, if MTBCF has not been calculated) is used as the total operating time, the study will show the impacts on bare-base substainability when deferred repair concepts are employed.

The sensitivity of MCSP to various system changes for the architecture and mission of Section 5.1 is shown in Table 10. Adding a second signal processor to Digital A reduces the number of mission failures by 55%. This change is entered in the architecture file by changing the number of branches in pool 14, chain 2 from one to two. Adding a third L-band receiver and switch port requires more file modifications: A separate pool is created for the L-band antenna connector; pool 10 is modified to have three branches, each containing one receiver and switch port; and the GPS utilization for pool 10 is changed to two. Allowing GPS to use Digital B is accomplished by adding GPS (function 1) to the function list

Table 10. Sensitivity of MCSP to Configuration Changes

Configuration Option	MCSP[a]	Δ(1-MCSP) (%)
Add signal processor to Digital A	0.99946	-55
Add preprocessor to Digital A	0.99880	0
Add L-band receiver and switch port	0.99892	-10
Add switching to allow GPS to use Digital B	0.99910	-25
Isolate the signal processor in each LRU	0.99880	0

[a]Baseline MCSP = 0.99880.

and adding GPS utilization rates for chain three. Isolating the signal processors is accomplished by changing their pool type to C in both pools. As this leaves no type S pools, the type F pools may also be changed to type N. The result that some architecture changes have no perceptible effect on MCSP may be counterintuitive. This insensitivity arises when the architecture change affects mission success only in the rare event of several specific resource failures. Adding a preprocessor to Digital A, for example, is only helpful if two preprocessors fail in that LRU. The sensitivity to these changes may increase if the total operating time is increased.

6. ADVANCED APPLICATIONS

This chapter provides suggestions on applying MIREM to more complex systems that do not fit readily into the model framework defined in Chapter 2. Special procedures and approximation techniques will be provided. These suggestions are based on experience gained in applying MIREM to the ICNIA system definition architectures.

6.1 Resource Pool Approximations

The methodology for identifying resource pools given in Section 3.5 does not apply exactly to some architectures. The following architecture design peculiarities may be encountered:

1. Parallel branches within a pool are not all the same.

2. Pool boundaries (parallel structures) differ for different functions.

3. Type C pools in parallel chains do not occur in pairs.

6.1.1 Branches Differ

If a pool contains branches with different failure rates, a good approximation is to use the average branch failure rate. It may be necessary to change the resource data to define a resource with the desired failure rate.

6.1.2 Pool Boundaries Differ

Pool boundaries can differ across functions because of switching limitations. For example, if the low-band LRU in

Figure 7 were connected to only two preprocessors in Digital A, these two preprocessors would be a pool with respect to UHF. These preprocessors would then be entered twice in Table 4, once in a three-branch structure for GPS and once in a two-branch structure for UHF and SINCGARS. However, MIREM assumes that pool boundaries are the same for all functions. This situation can be approximated by forming a pool from the set of branches with the least redundancy and putting any leftover branches in a separate pool. For the mission defined in Figure 32, three branches are required from the three-branch structure (GPS plus SINCGARS), and one branch is required from the two-branch structure (UHF or SINCGARS in separate phases). Therefore, a three-branch pool should be formed since it contains no redundancy. This formulation is approximate in that it allows low-band functions to use preprocessor 3. Forming two pools would mean that GPS must use preprocessor 3 and either preprocessor 1 or preprocessor 2.

Pool boundaries can also differ across functions with regard to how many resources are on a branch. In this case, the larger set of resources should be used in MIREM to define the branches.

6.1.3 Singleton Type C Pools

MIREM assumes that type C pools in parallel chain pairs occur in pairs, one in each chain. If a singleton pool occurs, add a pool with the same number of branches and utilizations to the other chain. Define a dummy resource with zero failure rate to put in the added pool.

6.2 Modeling Resource Scheduling with Utilization Rates

The methodology for assigning pool types and utilizations presented in Section 3.5 is intended to specify the number of branches required in a pool to perform each combination of functions. However, for systems that share resources between functions according to a complex schedule, the methodology may not apply. Pools may be contending with respect to some functions and noncontending with respect to others. Several techniques can be used to model resource scheduling.

The most desirable approach is to label ambiguous pools as type C and then define fractional utilizations in such a way that noncontending functions add correctly. One such situation, and the utilizations used to model it, is shown in Table 11. For more complex situations, it may be necessary to define two utilizations for each function: one to be used if a key function

Table 11. Fractional Utilizations for a Partially Contending Resource Pool

Functions Required			Total Utilization
1	2	3	
X			0.1 (1)
	X		0.1 (1)
		X	2.0 (2)
X	X		0.2 (1)
X		X	2.1 (3)
	X	X	2.1 (3)
X	X	X	2.2 (3)

is required, the other if it is not. These two utilizations can be stored in separate architecture files and used for the appropriate missions. In the case of a series chain, this practice can be taken as far as necessary, even to the point of defining new utilizations for each mission phase ("functions" then become mission phases). However, on parallel chains, the total utilization for each subset of the required functions is needed to correctly compute reliability; therefore, the technique should be used with caution.

Another situation that can be resolved through utilization rates is a shared pool pair that is utilized in a non-contending fashion. MIREM assumes that all type S pools are contending (utilizations add). The desired effect can be achieved by setting all utilizations to 0.01, except possibly for one function that requires more than one branch - reduce its utilization from n to n - 0.5.

6.3 Resource Chain Approximations

The methodology for identifying resource chains given in Section 3.4 does not apply exactly to some architectures. The following peculiarities may be encountered:

1. More levels of redundancy.

2. Chain boundaries (top-level switching points) differ for different functions.

6.3.1 General Series/Parallel Structures

If more than two levels of redundancy occur in the system or more than two redundant paths occur at the higher level (more than two parallel chains), the group feature of Section 2.3.2 can be used to create a general series/parallel structure. However, this prevents the feature of contention between functions (e.g., dedicated resources or contention for throughput of a data processor) and resource sharing from being modeled. When these features are significant, the lowest level of redundancy in the system may have to be neglected to allow modeling in MIREM.

6.3.2 Chain Boundaries Differ

If pools belong to different chains with respect to different functions (different switching points), an approximate model can be formed by placing these pools in the parallel chains instead of a series chain, or in the larger (more contribution to MCSP) parallel chains instead of the smaller ones.

APPENDIX A: MIREM EQUATIONS

This appendix describes the equations and algorithms used in the MIssion REliability Model (MIREM). The model's basic function is to evaluate the combinations of failures which result in failure of a particular mission and compute the probability of such failures. Intrinsic hardware reliability is not predicted by the model but is treated as an input. The general problem is defined in Section A.1, and the special class of problems that will be solved is presented in Section A.2. Reliability is computed in Sections A.3 and A.4. Some additional model outputs are derived in Sections A.5, A.6, and A.9. Section A.7 generalizes the problem to consider imperfect switching; and Section A.8, to consider repair.

A.1 The Phased Mission Reliability Problem

Assume that the system consists of n resources or "failure units" with constant failure rate. The traditional approach is to represent system health by $\underline{X}$, where X_i is equal to 1 if component i is good at the end of the mission, and 0 otherwise. For each mission M, the system structure function

$$\phi_M(\underline{X}) = \begin{cases} 1 \text{ if the mission M can be supported with system health } \underline{X} \\ 0 \text{ otherwise} \end{cases}$$

is determined and MCSP is just $\Pr\{\phi_M(\underline{X}) = 1\}$. Define a phased mission with m phases by letting X_i^{ℓ} equal 1 if component i is up at the end of phase ℓ and 0 otherwise, and let $\phi^{\ell}(\cdot)$ be the structure function for phase ℓ. Then

$$\text{MCSP} = \prod_{\ell=1}^{m} \Pr\{\phi^{\ell}(\underline{X}^{\ell}) = 1 \mid \phi^{h}(\underline{X}^{h}) = 1,\ h=1, \ldots, \ell-1\}$$

$$\geq \prod_{\ell=1}^{m} \Pr\{\phi^{\ell}(\underline{X}) = 1\} \qquad (1)$$

Even in the single-phase case, this approach is practical only if the system has few components or has a special modular structure. Furthermore, in order to analyze various mission requirements, it is desirable to express ϕ^{ℓ} at the individual function level rather than for a phase. Phases with various function requirements can then be formulated if a "combining" operation is defined on the functional structures.

A.2 A Special Structure for Integrated Reconfigurable Electronics

For the reasons discussed above, ϕ^{ℓ} will not be dealt with explicitly. Instead, the special structure of ϕ^{ℓ} which has been observed in advanced avionics architectures will be exploited to allow more efficient computations.

Chain Structures - Assume that the system can be described by either a one-level or two-level structure. A one-level structure consists of a set of k-of-n modules in series. These k-of-n modules will be referred to as pools. Each parallel branch in a pool contains one or more resources in series (branches are identical). The number of branches (k) required in a pool depends on the function requirements. Pools that are irrelevant (i.e., k equal to 0) with respect to certain functions are allowed.

A two-level structure consists of a set of one-level structures. Each one-level structure will be referred to as a chain. Chains are either in "series" in the sense that all functions must use the chain, or "parallel" in the sense that a set of functions is supportable if there exists an allocation of functions to parallel chains such that each chain can support its functions. Some functions may be restricted to certain chains. Note that parallel chains are not modules (i.e., not a series-parallel structure). Instead of simply up or down, these chains must be described by the combinations of functions which they can support. The current implementation of MIREM allows only two chains in parallel (chain pairs), although generalization is possible.

A slight generalization to this model is also considered. The allocation of functions to parallel chains may not be strict in that pools may be shared between parallel chains. For example, processing resources in parallel chains may be shared if the resources communicate through data buses.

Series-Parallel Structures - General series-parallel structures, with more than the two levels described above, will

also be considered by organizing pools into k-of-n structures within a series chain.

A.3 Single-Chain Computations

For pools in a single chain, let

$C_i(t)$ = number of good branches in pool i at time t

u_{ij} = utilization of pool i by function j

$c_{max,i}$ = number of branches in pool i (no failures)

Now define two types of pools, according to how functions combine. If a set F_ℓ of functions is required in phase ℓ, the total requirement for pool i in phase ℓ is

$$r_{i\ell} = \begin{cases} \sum_{j \varepsilon F_\ell} u_{ij} & \text{if pool i is contending} \\ \max_{j \varepsilon F_\ell} u_{ij} & \text{if pool i is noncontending} \end{cases} \qquad (2)$$

Also let

$$r_i = \max_{\ell=1,\ldots,m} r_{i\ell} \qquad (3)$$

If the functions are not required simultaneously, all pools are considered noncontending.

Assume that the system is initially fully repaired, so that $\Pr\{C_i(0) = c_{max,i}\} = 1$, and that the time until failure of each resource is exponentially distributed. Then for active redundant pools, the exponential failure time distribution implies that

$$\Pr\{C_i(t) = k\} = \binom{c_{max,i}}{k} (p)^{c_{max,i}-k}(1-p)^k, \quad k \leq c_{max,i} \qquad (4)$$

where

$p = e^{-\lambda t}$

t = operating time

λ = branch failure rate (sum of resource failure rates on a branch)

For standby redundant pools, only the required number of branches in each phase, approximated by r_i, are subject to failures:

$$Pr\{C_i(t) = k\} = (r_i \lambda t)^{c_{max,i}-k} e^{-r_i \lambda t} / (c_{max,i}-k)! \qquad (5)$$

Using the approximation of Equation 1, MCSP for a single chain is just

$$MCSP = \prod_{pool\ i} Pr\{C_i(t) \geq r_i\} \qquad (6)$$

which can be easily computed from Equations 4 or 5.

If the chain is cascading (that is, it contains groups of noncontendings pools in k-of-n structures), the reliability is computed by starting with the smallest groups and working out. Let

n_s = *number of subgroups (pools or groups) contained in the group*

p_j = MCSP (reliability) of subgroup j, $j=1,..,n_s$

r = requirement for the group (computed in the same manner as for noncontending pools)

i_j = 0 or 1, $j=1,\ldots,n_s$

$I = [i_j]$

$$|I| = \sum_{j=1}^{n_s} i_j$$

The group MCSP is

$$MCSP_{GROUP} = \sum_{|I| \geq r} \prod_{j=1}^{n_s} p_j^{i_j}(1-p_j)^{1-i_j} \qquad (7)$$

By defining the largest group to contain the whole chain, the last group MCSP computed will be the chain MCSP.

A.4 Parallel Chain Computations

Now consider a two-level structure containing two parallel chains. Pools are divided into the following pool types:

- F: chain-fail pools (noncontending)
- S: shared pools (contending)
- N: noncontending pools, excluding types F and S
- C: contending pools, excluding types F and S

All type S and type C pools in parallel chains must occur in pairs, one in each chain. Pool pairs must have the same function utilization rates. A pair of pools, one in each chain, is type S if each pool's resources can be used by functions allocated to the opposite chain. Type F pools are those which, upon failure, prevent the entire chain (including type S pools) from being utilized. The remaining pools are classified as type N or type C according to Equation 2. For the requirement calculation of Equation 2, type F pools are treated like type N, and type S pools are treated like type C. If functions are not required simultaneously, type C and type S pools are treated as type N.

Define the following function sets:

$$CF = \bigcup_{\ell=1}^{m} F_\ell$$

CFCOM = {functions in CF that can use either chain in the pair}

CF^k = {functions in CF that must use chain k}

$FCOM_\ell$ = {functions in F_ℓ that can use either chain in the pair}

F_ℓ^k = {functions in F_ℓ that must use chain k}

for k=1 (primary chain) and k=2 (secondary chain). The state of a chain as determined by its C_i implies the ability to support certain functions. Let

$$X_j^k = \begin{cases} 1 & \text{if function j can be supported after operating for time t on the type N pools on chain k,} \\ 0 & \text{otherwise} \end{cases}$$

$\underline{X}^k$ = $[X_j^k]$, $j \varepsilon CFCOM$

$UP^k(\alpha)$ = the event that the set of functions CF can be supported after operating for time t on the type α pools on chain k

$UP^{1+2}(\alpha)$ = the event that the set of functions CF can be supported after operating for time t on the type α pools on the pair of parallel chains

for k=1 (primary) and k=2 (secondary) and α = F, S, N, C. The event $UP^{1+2}(C)$ is dependent upon $\underline{X}^k$ in that an allocation of functions to chains that is supportable on the type C pools must be consistent with the supportability of functions on the type N pools. Similarly, the event $UP^{1+2}(S)$ is dependent on $UP^k(F)$. Applying these definitions and using the phase approximation of Equation 1,

$$\begin{aligned} MCSP &= Pr\{UP^{1+2}(F,S,N,C)\} \\ &= Pr\{UP^{1+2}(C) | UP^{1+2}(N)\} \cdot Pr\{UP^{1+2}(N)\} \\ &\cdot Pr\{UP^{1+2}(S) | UP^1(F), UP^2(F)\} \cdot Pr\{UP^1(F)\}\ Pr\{UP^2(F)\} \\ &+ Pr\{UP^1(S,N,C)\} \cdot Pr\{UP^1(F)\} \cdot [1 - Pr\{UP^2(F)\}] \\ &+ Pr\{UP^2(S,N,C)\} \cdot Pr\{UP^2(F)\} \cdot [1 - Pr\{UP^1(F)\}] \end{aligned} \qquad (8)$$

The three terms in Equation 8 correspond to both chains being up with respect to type F pools, chain 1 being up and chain 2 being up. Two algorithms have been developed to compute the first term.

A.4.1 Quick Algorithm

This approximate algorithm will work for most problems. In the current software, the size of CFCOM is limited to eight. Conditioning on $\underline{X}^k$,

$$\Pr\{UP^{1+2}(C) \mid UP^{1+2}(N)\} \cdot \Pr\{UP^{1+2}(N)\}$$

$$= \sum_{\underline{x}^1+\underline{x}^2 \geq \underline{1}} \Pr\{UP^{1+2}(C) \mid \underline{X}^1 = \underline{x}^1, \underline{X}^2 = \underline{x}^2\} \Pr\{\underline{X}^1 = \underline{x}^1\} \Pr\{\underline{X}^2 = \underline{x}^2\} \tag{9}$$

The distribution of $\underline{X}^k$ is determined by applying the single-chain analysis of Section A.3 to the type N pools for all subsets of the functions CFCOM (the functions CF^1 are always required on chain 1 and the functions CF^2 are always required on chain 2), giving $\Pr\{\underline{X}^k \geq \underline{x}\}$ for all $\underline{x}$. The law of total probability is then used to obtain $\Pr\{\underline{X}^k = \underline{x}\}$.

The type C pools are treated as follows. Assume that type C pools occur in pairs, one on each chain, and use the index i to refer to pairs rather than individual pools. A superscript will be used to indicate chain (e.g., C_i^k, $c_{max,i}^k$). The utilizations u_{ij}, however, are assumed to be the same for both chains. The allocation of functions to chains is represented by

$$y_j = \begin{cases} 1 & \text{if function j uses chain 1} \\ 0 & \text{if function j uses chain 2} \end{cases}$$

$$\underline{y} = [y_j] \ , \quad j \varepsilon CFCOM$$

Let

$$r_{i,\ell}(\underline{y}) = \sum_{j \in FCOM_\ell} y_j \, u_{ij} \tag{10a}$$

$$r^k_{i,\ell} = \sum_{j \in F^k_\ell} u_{ij} \tag{10b}$$

$$r_i = \max_{\ell=1,\ldots,m} \left\{ r_{i,\ell}(\underline{1}) + r^1_{i,\ell} + r^2_{i,\ell} \right\} \tag{10c}$$

$$r_{max,i} = \max_{j \in CFCOM} u_{ij} \tag{10d}$$

for k=1,2. The conditional event in Equation 9 occurs if there exists an allocation $\underline{y}$ such that

$$\underline{1} - \underline{x}^2 \leq \underline{y} \leq \underline{x}^1 \tag{11a}$$

$$\max_{\ell=1,\ldots,m} \left\{ r_{i,\ell}(\underline{y}) + r^1_{i,\ell} \right\} \leq C^1_i \tag{11b}$$

$$\max_{\ell=1,\ldots,m} \left\{ r_{i,\ell}(\underline{1} - \underline{y}) + r^2_{i,\ell} \right\} \leq C^2_i \tag{11c}$$

for all type C pools i. That is, functions can be assigned only to chains on which the type N pools can support them, and the total function requirements in each phase must not exceed the type C pool capacities.

A <u>necessary</u> condition for such an allocation to exist is

$$\max_{\ell=1,\ldots,m} \left\{ r_{i,\ell}(\underline{1} - \underline{x}^2) + r^1_{i,\ell} \right\} \leq C^1_i \tag{12a}$$

$$\max_{\ell=1,\ldots,m} \left\{ r_{i,\ell}(\underline{1} - \underline{x}^1) + r^2_{i,\ell} \right\} \leq C^2_i \tag{12b}$$

$$r_i \leq C_i^1 + C_i^2 \tag{12c}$$

$$r_{max,i} \leq \max \{C_i^1, C_i^2\} \tag{12d}$$

for all type C pools i. The probability of condition 11 will be approximated by the probability of condition 12c. To motivate this approximation, note that condition 12c requires that sufficient resources be available to perform the required functions in each phase. Hence, errors occur in this approximation only in calculating the probability that the required resources are divided in usable proportions on the two chains. There are two reasons why condition 12c may not be sufficient:

1. Discrete allocation - there may be no division of $FCOM_\ell$ that matches the available resources in each chain.

2. Lockout - the available resources in pool pair i may be on the wrong chain; i.e., functions are "locked out" of that chain because they cannot be supported by other type C pools or by type N pools.

The discrete allocation problem, which exists only if u_{ij} takes on different or noninteger values, is treated by condition 12d, which ensures that the largest u_{ij} can be allocated. This approximation will generally be very good when there is at most one dominant u_{ij} (one more demanding function). The lockout problem for type N pools is treated exactly by conditions 12a-b; functions are allocated only to chains on which they can be supported by the type N pools. If there is more than one type C pool pair, lockout between these resources will not be captured. Hence, the approximation will be good if there are few "problem" type C pools and will be optimistic if there are a number of such pools.

Using condition 12,

$$\Pr\{UP^{1+2}(C) \mid \underline{X}^1 = \underline{x}^1, \underline{X}^2 = \underline{x}^2\}$$

$$\doteq \prod_{\substack{\text{type C} \\ \text{pools } i}} \sum_{c^1=h}^{c^1_{max,i}} \Pr\{C_i^1 = c^1\} \cdot \Pr\{C_i^2 \geq c^2\} \tag{13}$$

where

$$h = \max_{\ell=1,\ldots,m} \{r_{i,\ell}(\underline{1}-\underline{x}^2) + r^1_{i,\ell},\ r_i - c^2_{max,i}\}$$

$$c^{2'} = \max_{\ell=1,\ldots,m} \{r_{i,\ell}(\underline{1}-\underline{x}^1) + r^2_{i,\ell},\ r_i - c^1_{max,i}\}$$

$$c^2 = \begin{cases} c^{2'} & \text{if } c^1 \geq r_{max,i}\ , \\ \max\{c^{2'},\ r_{max,i}\} & \text{otherwise} \end{cases}$$

A.4.2 Full Algorithm

This exact algorithm will work for almost all single-phase problems but cannot be used with multiple phases. Hence, phase subscripts will be dropped during this discussion. Continuing along the lines of Equation 10a-b, the total requirement in pool i under allocation $\underline{y}$ is

$$r_i^{TOT}(\underline{y}) = \begin{cases} \sum_{j\varepsilon FCOM} y_j\, u_{ij} + \sum_{j\varepsilon F^1} u_{ij} & \text{, type C pool in chain 1} \\ \sum_{j\varepsilon FCOM} (1-y_j)u_{ij} + \sum_{j\varepsilon F^2} u_{ij} & \text{, type C pool in chain 2} \\ \max_{j:\ j\varepsilon FCOM \text{ and } y_j=1,\ \text{or } j\varepsilon F^1} u_{ij} & \text{, type N pool in chain 1} \\ \max_{j:\ j\varepsilon FCOM \text{ and } y_j=0,\ \text{or } j\varepsilon F^2} u_{ij} & \text{, type N pool in chain 2} \end{cases}$$

If allocations $\underline{y}$ and $\underline{z}$ satisfy $r_i^{TOT}(\underline{y}) = r_i^{TOT}(z)$ for all type N and type C pools i, then they will be called substantively identical allocations. Define

A = {allocations that are not substantively identical}

n_o = order of A

$A(k)$ = {all order k subsets of A}

Allocation $\underline{y}$ is possible if it can be supported, $C_i \geq r_i^{TOT}(\underline{y})$, on all type N and type C pools. The event $UP^{1+2}(N,C)$ corresponds to at least one allocation being possible. Denote an element of A(k) by $a = (\underline{y}^1, \ldots, \underline{y}^k)$, and let

$p(a) = Pr\{y^1, \ldots, y^k \text{ are all possible allocations}\}$

Then the law of total probability gives

$$Pr\{UP^{1+2}(N,C)\} = \sum_{k=1}^{n_o} (-1)^{k-1} \prod_{a \varepsilon A(k)} p(a) \qquad (14)$$

The current MIREM software limits n_o to 15, which corresponds to about 32,000 terms in Equation 14. Double-precision arithmetic is used to reduce the effect of roundoff error. The evaluation of p(a) has the same form as a series chain computation:

$$p(a) = \prod_{\substack{\text{pool } i \\ \text{type N or C} \\ \text{in chain 1 or 2}}} Pr\{C_i(t) \geq \max\{r_i(\underline{y}^1), \ldots, r_i(\underline{y}^k)\}\} \qquad (15)$$

A.4.3 System Reliability

Regardless of whether the quick or full algorithm is used, the type S pools are treated as follows. Assume that type S pools occur in pairs, one on each chain, and use the same notation as for type C pools. Because the paired resources are shared, only the combined capacity of the two pools need be considered:

$$\Pr\{UP^{1+2}(S) \mid UP^1(F),\ UP^2(F)\} = \prod_{\substack{\text{type S}\\ \text{pools } i}} \Pr\{C_i^1 + C_i^2 \geq r_i\}$$

$$= \prod_{\substack{\text{type S}\\ \text{pools } i}} \sum_{c^1=h}^{c^1_{\max,i}} \Pr\{C_i^1 = c^1\}\ \Pr\{C_i^2 \geq r_i - c^1\} \qquad (16)$$

where

$$h = \max_{\ell=1,\ldots,m} \{r_{i,\ell}(\underline{1}) - c^2_{\max,i}\}$$

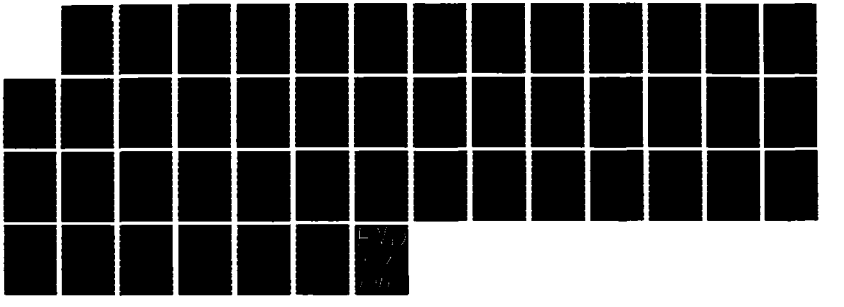
Applying the single-chain analysis to the type F pools gives $\Pr\{UP^k(F)\}$. This completes the evaluation of the first term of Equation 8.

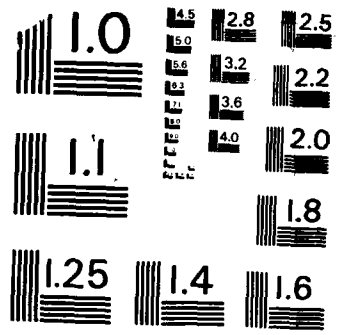
To evaluate the second and third terms, only $\Pr\{UP^k(S,N,C)\}$ is needed. It is obtained by applying the single-chain analysis to the type S, type N, and type C pools using the set of functions F_ℓ for phase ℓ. Note that if not all functions in CF are supported on chain k ($CF^1 \neq \emptyset$ for k=2 or $CF^2 \neq \emptyset$ for k=1), then $\Pr\{UP^k(S,N,C)\} = 0$.

Equation 8 gives MCSP for a pair of parallel chains. If the system contains several chains or parallel chain pairs in series, with reliabilities $MCSP_i$, the combined reliability is

AD-A175 235 MISSION RELIABILITY MODEL USERS GUIDE(U) ANALYTIC SCIENCES CORP READING MA M H YEATCH ET AL. NOV 86 AFHRL-TR-86-35 F19628-82-C-0002 2/2

UNCLASSIFIED F/G 14/4 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS - 1963 - A

$$MCSP = \prod_{\substack{\text{chain} \\ \text{pairs} \\ i}} MCSP_i \qquad (17)$$

A.5 Phase Sequence Algorithm

An algorithm is presented that computes upper and lower bounds on the probability that each mission phase is successfully completed. For phase ℓ, this probability is

$$MCSP_\ell = \prod_{h=1}^{\ell} Pr\{\phi^h(\underline{X}^h)=1 \mid \phi^k(\underline{X}^k)=1,\ k=1,\ldots,h-1\} \qquad (18)$$

The $MCSP_\ell$ bounds culminate in MCSP bounds for the entire mission after the final phase m. The calculations in Sections A.3 and A.4 give a different lower bound on MCSP because of the assumption in Equation 2 that all phases are required at the end of the mission; that approximation is not made here.

Let

t_ℓ = duration of phase ℓ (hours)

$T_\ell = \sum_{h=1}^{\ell} t_h$ = cumulative time after ℓ phases

$MCSP_\ell^L$ = lower bound for $MCSP_\ell$

$MCSP_\ell^U$ = upper bound for $MCSP_\ell$

Phase ℓ is said to be dominated by phase h if the event that the functions F_h are available implies that the functions F_ℓ are available. A sufficient condition for phase ℓ to be dominated by phase h is $F_\ell \subseteq F_h$. This condition will be used to test for domination.

In the following algorithm,

ND = set of nondominated phases among phases $1,\ldots,\ell$

DT_ℓ = cumulative duration of phase ℓ plus earlier phases dominated by phase ℓ

Initialization. ND = $\emptyset$;

$MCSP_0^U = MCSP_0^L = 1$;

$\ell=1$; $T_0 = 0$.

1. D = {phases in ND that are dominated by phase ℓ}

2. $DT_\ell = t_\ell + \sum_{h\in D} DT_h$

3. P_ℓ^L = Pr{functions F_ℓ are available at time T_ℓ}

   P_ℓ^S = Pr{functions F_ℓ are available at time DT_ℓ}, computed by applying Sections A.3 and A.4 to a one-phase mission.

4. $MCSP_\ell^L = MCSP_{\ell-1}^L \cdot P_\ell^L$

5. $MCSP_\ell^U = MCSP_{\ell-1}^U \cdot P_\ell^S / \prod_{h\in D} P_h^S$

6. If $\ell=m$, then stop; $MCSP_\ell$ is for the whole mission.

7. Update ND = ND - D + $\{\ell\}$

8. $\ell = \ell+1$. Go to step 1.

The tightness of these bounds tends to decrease as more phases are considered.

A.6 Mean Time Between Critical Failure Algorithm

Another measure which can be computed by MIREM is MTBCF, defined as the expected operating time without repair until a critical function is lost, starting with full system capacity. Let F(t) be the MCSP for an operating time of t hours. Then

$$\text{MTBCF} = -\int_0^\infty t \, d\bar{F}(t)$$

$$= \int_0^\infty \bar{F}(t)dt \qquad (19)$$

This integral is evaluated in MIREM using the trapezoidal rule with a variable step size which can be modified by exploration.

1. Select α, δ, ε_{min}, ε_{rel} and t_1. Initialize $t_o = 0$, $dt = t_1 - t_o$, $k = 1$.

$$\lambda_1 = \ln[\bar{F}(t_o)/\bar{F}(t_1)]/dt$$

$$\Delta = [\bar{F}(t_o) + \bar{F}(t_1)]dt/2$$

$$\text{MTBCF} = \Delta$$

2. $dt(\delta) = 0.8\delta \, dt/(\Delta \, \lambda_k)$

3. If $k \leq 2$, $dt = \min\{\alpha \, dt, \, dt(\delta)\}$.

If $k > 2$, $dt(\varepsilon_{rel}) = \left| 8\bar{F}(t_{k-1})(0.8\varepsilon_{rel})/ \frac{d^2 F}{dt^2} \right|^{1/2}$

and $dt = \min\{\alpha \, dt, \, \max\{dt(\delta), \, dt(\varepsilon_{rel})\}\}$.

4. $k = k + 1$.

5. $t_k = t_{k-1} + dt$

$$\frac{d^2\bar{F}}{dt^2} = 2\left[\frac{\bar{F}(t_k)-\bar{F}(t_{k-1})}{t_k - t_{k-1}} - \frac{\bar{F}(t_{k-1})-\bar{F}(t_{k-2})}{t_{k-1} - t_{k-2}}\right] \Bigg/ (t_k - t_{k-2})$$

$$\lambda_k = \ln[\bar{F}(t_{k-1})/\bar{F}(t_k)]/dt$$

$$\Delta = [\bar{F}(t_{k-1}) + \bar{F}(t_k)]dt/2$$

6. If $\varepsilon_{rel} < (dt)^2 \frac{d^2\bar{F}}{dt^2} / [8\bar{F}(t_{k-1})]$

and $\delta < \Delta\ \lambda_k$, then set $dt = dt/2$ and go to step 5.

7. $MTBCF = MTBCF + \Delta$

8. If $\bar{F}(t_k)/\lambda_k > 0.1$

and $\varepsilon_{min} < \frac{|\lambda_k - \lambda_{k-1}|}{\lambda_k(t_k - t_{k-1})}$,

then go to step 2. Otherwise, set

$MTBCF = MTBCF + \bar{F}(t_k)/\lambda_k$

and stop.

This algorithm is based on the assumption that, at least for large t, F(t) can be approximated by $ae^{-\lambda t}$. Local estimates of λ serve as a basis for selecting a step size, $dt(\delta)$, which

will include the desired fraction δ of the entire integral. Estimates of λ are also used as a stopping criterion. If the relative change in λ is less than ε_{min} per unit change in t, it is assumed that the remainder of F has a constant failure rate and it is integrated analytically. The parameter ε_{rel} provides an alternative basis for increasing the step size, based on the average relative error in F calculated from its second derivative. The scaling parameter α sets a limit on how rapidly step size can increase.

The parameters values that are used in MIREM are

$$\alpha = 4$$

$$\delta = 0.025$$

$$\varepsilon_{min} = 0.00001 \text{ hrs}^{-1}$$

$$\varepsilon_{rel} = 0.005$$

$$t_1 = 0.025 \cdot \text{MTBF}$$

Tests indicate that the MTBCF accuracy obtained using these values is better than 0.5%, while an average of only 22 function evaluations was required. These results suggest that the algorithm is more efficient, for the life distributions considered, than a general-purpose routine.

A.7 Imperfect Switching

The previous sections have assumed that the system controller always selects a configuration in which the relevant functions are available, if such a configuration exists. This perfect switching assumption requires

(1) Perfect Built-In Test (BIT) fault isolation, and

(2) Optimal reconfiguration logic in the controller.

Condition (1) will now be relaxed to consider undetected failures and false alarms. Let

$$A_i = \begin{cases} 1 & \text{if branch i is } \underline{\text{actually}} \text{ operable} \\ 0 & \text{o.w.} \end{cases}$$

$$\underline{A} = [A_i]$$

$$B_i = \begin{cases} 1 & \text{if branch i is } \underline{\text{believed}} \text{ operable by the controller} \\ 0 & \text{o.w.} \end{cases}$$

$$\underline{B} = [B_i]$$

Ψ = {possible configurations}

$\psi(\underline{b})$ = the configuration selected when $\underline{B} = \underline{b}$, $\psi(\underline{b}) \in \Psi$

$$\phi(\underline{a}) = \begin{cases} 1 & \text{if there is a configuration in which the system is up when } \underline{A} = \underline{a} \\ 0 & \text{o.w.} \end{cases}$$

$$\Gamma(\underline{a},\psi) = \begin{cases} 1 & \text{if the configuration } \psi \text{ makes the system up when } \underline{A} = \underline{a} \\ 0 & \text{o.w.} \end{cases}$$

T = system "life"; the first time at which $\Gamma(\underline{A},\psi(\underline{B})) = 0$

T_B = believed system life; the first time at which $\phi(\underline{B}) = 0$

A mission of length t can have six outcomes, with probabilities $p_I, \ldots, p_{VI}$:

I. <u>Up and Believed Up</u>: $T, T_B > t$. At time t, $\Gamma(\underline{A}, \psi(\underline{B})) = 1$ and $\phi(\underline{B}) = 1$.

II. Up and Believed Down: $T_B < t < T$. At time t, $\Gamma(A,\psi(\underline{B})) = 1$ and $\phi(\underline{B}) = 0$.

III. Down and Believed Down: $T, T_B < t$ and, at time t, $\phi(\underline{A}) = 0$ (system would be down even with perfect switching). At time t, $\phi(\underline{B}) = 0$.

IV. Down and Believed Up: $T < t < T_B$ and, at time t, $\phi(\underline{A}) = 0$ (system would be down even with perfect switching). At time t, $\phi(\underline{B}) = 1$.

V. Wrong Configuration and Believed Down: $T, T_B < t$ and, at time t, $\phi(\underline{A}) = 1$ (system would be up with perfect switching). At time t, $\phi(\underline{B}) = 0$. At some time in $[0,t]$, $\Gamma(\underline{A},\psi(\underline{B})) = 0$.

VI. Wrong Configuration and Believed Up: $T < t < T_B$ and, at time t, $\phi(\underline{A}) = 1$ (system would be up with perfect switching). At time t, $\phi(\underline{B}) = 1$. At some time in $[0,t]$, $\Gamma(\underline{A},\psi(\underline{B})) = 0$.

These six events are mutually exclusive and exhaustive; their relationship is shown in Figure A-1. Reliability of the imperfect switching system is

$$MCSP_{IMP} = p_I + p_{II} \tag{20}$$

Reliability of the perfect switching system,

$$MCSP_{PERF} = p_I + p_{II} + p_V + p_{VI} \tag{21}$$

can be evaluated using Section A.4. If the mission is aborted when the system is believed down, mission reliability is p_I.

A.7.1 False Aborts

A mission abort, or believed failure, occurs without an actual system failure with probability p_{II}. First, an upper bound for p_{II} will be derived.

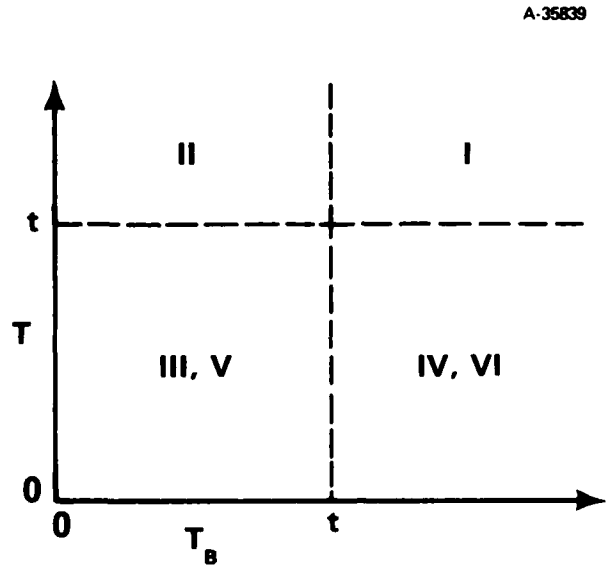


Figure A-1. System Life Outcomes.

Define the additional (0,1) resource status vectors:

$\underline{U}_L$ = latent undetected failures at time t (branch has not been used since the undetected failure)

$\underline{U}_M$ = manifest undetected failures at time t (branch has been used since the undetected failure)

$\underline{U} = \underline{U}_L + \underline{U}_M$ = undetected failures

$\underline{F}$ = false alarms

$|\underline{x}| = \sum x_i$

Then

$$\underline{B} = \underline{A} - \underline{U} + \underline{F} \tag{22}$$

Also define

η_i = undetected failure rate in pool i

α_i = false alarm failure rate in pool i

δ_i = detected failure rate in pool i

$\underline{\eta} = [\eta_i]$

$\underline{\alpha} = [\alpha_i]$

$\underline{\delta} = [\delta_i]$

$$\eta = \sum_i \eta_i \cdot \text{(no branches in pool i)}$$

$$\alpha = \sum_i \alpha_i \cdot \text{(no branches in pool i)}$$

$r_i(k)$ = pool i requirement under allocation (of functions to chains) k

$R_\lambda(t)$ = reliability of perfect switching system at time t with pool failure rates $\underline{\lambda}$.

Now, since event II can occur only if these are false alarms,

$$P_{II} = Pr\{T_B < t < T\}$$

$$= Pr\{T_B < t \text{ and } \underline{F} \neq 0\}\ Pr\{T > t \mid T_B < t, \underline{F} \neq 0\} \tag{23}$$

$$Pr\{T_B < t \text{ and } \underline{F} \neq 0\} = Pr\{T_B < t\} - Pr\{T_B < t \mid \underline{F} = 0\}\ Pr\{\underline{F} = 0\}$$

$$= 1 - R_{\underline{\alpha}+\underline{\delta}}(t) - [1 - R_{\underline{\delta}}(t)]e^{-\alpha t} \tag{24}$$

It will be assumed throughout that false alarms increase the chance of mission failure since, for any system, some allocation

scheme ψ satisfies this assumption. Then the second term of Equation 23 can be bounded by

$$\begin{aligned} \Pr\{T > t \mid T_B < t, \underline{F} \neq 0\} &\leq \Pr\{T > t\} \\ &\leq \Pr\{T > t \mid \underline{F} = 0\} \\ &= \Pr\{\phi(\underline{A})=1, \underline{U}_M = 0 \mid \underline{F} = 0\} \\ &\leq \min\{R_\delta(t), \Pr\{\underline{U}_M = 0\}\} \end{aligned} \tag{25}$$

The probability of a single undetected failure being used during a mission is at least as great as the probability that the branch is in use when it fails. This probability may depend on the allocation; a lower bound is

$$\rho = \min_{\substack{\text{allocations} \\ \ell}} \rho(\ell) \tag{26}$$

where

$$\rho(\ell) = \frac{1}{\eta} \sum_{\text{pools } i} \eta_i \, r_{i\ell}$$

$r_{i\ell}$ = pool i requirement under allocation ℓ.

Hence,

$$\Pr\{\underline{U}_M = 0 \mid |\underline{U}| = 1\} \leq 1 - \rho \tag{27}$$

Assuming that no single branch dominates the undetected failures ($\eta_i << \eta$), a Poisson approximation to the failure process will be accurate:

$$\Pr\{\underline{U}_M = 0\} = \sum_{n=0}^{\infty} \Pr\{|\underline{U}_M| = 0 \mid |\underline{U}| = n\} \Pr\{|\underline{U}| = n\}$$

$$\leq \sum_{n=0}^{\infty} (1-\rho)^n e^{-\eta t} (\eta t)^n/n!$$

$$= e^{-\rho\eta t} \qquad (28)$$

Combining Equations 25 and 28,

$$\Pr\{T > t \mid T_B < t, \underline{F} \neq 0\} \leq \min \{R_\delta(t), e^{-\rho\eta t}\} \qquad (29)$$

Additional bounds for p_{II} can be obtained by combining bounds on $p_I + p_{II}$ from above with the following bounds on p_I. Since $T < t < T_B$ implies $\underline{U} \neq 0$,

$$p_I = \Pr\{T > t \text{ and } T_B > t\}$$

$$\geq \Pr\{T_B > t\} \Pr\{\underline{U} = 0\}$$

$$= R_{\underline{\alpha}+\underline{\delta}}(t)\, e^{-\eta t} \qquad (30)$$

Also, using Equation 28,

$$p_I = \Pr\{T_B > t\} \Pr\{\underline{U}_M = 0 \mid T_B > t\}$$

$$\leq R_{\underline{\alpha}+\underline{\delta}}(t)\, e^{-\rho\eta t} \qquad (31)$$

A.7.2 Mission Failures

The contribution of imperfect switching to mission failures is $MCSP_{PERF} - MCSP_{IMP}$. Approximate bounds are

$$MCSP_{IMP} = Pr\{T > t\}$$

$$\leq Pr\{T > t \mid \underline{F} = 0\}$$

$$= R_{\underline{\delta}}(t)\ Pr\{\underline{U}_M = 0 \mid \phi(\underline{A}-\underline{U}) = 1,\ \underline{F} = 0\}$$

$$\leq R_{\underline{\delta}}(t)\ e^{-\rho\eta t} \qquad (32)$$

and

$$MCSP_{IMP} \doteq Pr\{T > t \mid \underline{F} = 0\}$$

$$\geq R_{\underline{\delta}}(t)\ Pr\{\underline{U} = 0\}$$

$$= R_{\underline{\delta}}(t)\ e^{-\eta t} \qquad (33)$$

$MCSP_{PERF}$ is known exactly; it is just $R_{\underline{\delta}+\underline{\eta}}(t)$.

A.8 Repair Policies

In this section, the analysis is extended to consider repair. Four repair policies are considered:

1. Immediate Repair - repair all faults at the end of each mission.

2. Deferred Repair - repair when a critical failure occurs.

3. Scheduled Maintenance - repair after a specified operating time or when a critical failure occurs.

4. Repair at Degraded Level - repair when the number of branches available in some pool falls below a specified minimum.

The MCSP calculation of Sections A.3 and A.4 assumes that the system is fully repaired at the beginning of the mission; this is consistent with immediate repair. The MTBCF calculation of Section A.5 assumes that no repair occurs before critical failure; this is consistent with deferred repair. For each repair policy, the following additional quantities will be derived:

$\overline{MCSP}$ = average MCSP (over all possible states of repair) for an aircraft being maintained by the given repair policy

MTBMA = mean time between maintenance action

MTTR = mean time to repair the system

$$A_I = \frac{MTBMA}{MTBMA+MTTR} = \text{inherent availability}$$

Consider a system whose state of repair has reached steady state. Then

$$MTBCF^{-1} = \frac{E[\text{Number of critical failures in } (0,t)]}{t}$$

$$\geq \frac{Pr\{1 \text{ or more critical failures in } (0,t)\}}{t}$$

$$= \frac{(1 - \overline{MCSP})}{t}$$

Also, because a system composed of constant failure rate components has an increasing failure rate, the rate of critical failure goes down after one occurs and the system is repaired. Therefore, the probability of two or more critical failures is bounded above by a Poisson distribution with rate -ln MCSP. Because the rate parameter is also the mean,

$$MTBCF \geq t/(-\ln \overline{MCSP}) \qquad (34)$$

It can be shown that Equation 34 is accurate to within t, regardless of when failures occur within a mission. It will be used to relate MTBCF and $\overline{MCSP}$.

A.8.1 MTTR

Let

N_F = number of failures repaired per maintenance action

$MTTR(j)$ = MTTR of resource type j

n_j = number of type j resources in the system

λ_j = failure rate of resource j

MTBF = mean time between failure, critical or noncritical

$MTTR_{RES}$ = average resource MTTR.

Neglecting standby pools and unrepaired failures,

$$MTBF = \left(\sum_j n_j \lambda_j\right)^{-1} \tag{35a}$$

$$MTTR_{RES} = MTBF \cdot \sum_j n_j \lambda_j MTTR(j) \tag{35b}$$

$$E[N_F] = MTBMA/MTBF \tag{35c}$$

Under a series repair concept (one repairman), there is no availability payoff for the deferral of repairs (except for the second-order effect of a reduced number of failures, which is neglected below):

$$MTTR = E[N_F]\ MTTR_{RES}$$

$$= MTBMA\ (MTTR_{RES})/MTBF \tag{36a}$$

$$A_I = \frac{MTBMA}{MTBMA + MTTR}$$

$$= \frac{MTBF}{MTBF + MTTR_{RES}} \tag{36b}$$

To analyze a parallel repair concept (unlimited repairmen), let

$F(t)$ = distribution function of MTTR for one resource failure

The system repair time is the maximum of the repair times for the failed resources. Given the number of failures, the mean (MTTR) can be approximated as a percentile of the distribution function $F(t)$; that is, MTTR satisfies

$$F(MTTR) = \alpha \tag{37}$$

for some percentile $0 < \alpha < 1$. A "typical," uniform distribution of N_F failures gives percentiles of $1/(N_F+1), \ldots N_F/(N_F+1)$; i.e., a maximum of $N_F/(N_F+1)$. In principle, one could take the expectation of MTTR over N_F using this percentile. For simplicity, interpolate and use

$$\alpha = E(N_F) \,/[E(N_F)-1] \tag{38}$$

in Equation 37 to approximate MTTR.

A.8.2 Immediate Repair

Under immediate repair $\overline{MCSP}$ = MCSP (calculated in Sections A.3 and A.4). Also, MTBMA = MTBF since all failures are repaired.

A.8.3 Deferred Repair

Under deferred repair, MTBCF is computed using Section A.6 and converted to MCSP using Equation 34. Also, MTBMA = MTBCF since only critical failures are repaired.

A.8.4 Scheduled Maintenance

For scheduled maintenance, let

T_M = operating time between scheduled maintenance

$n_M = \lceil T_M/t \rceil$ = number of missions between scheduled maintenance

$\lceil x \rceil$ = smallest integer greater than or equal to x.

τ = operating time at which critical failure occurs

$\tau_i = E[\tau \mid (i-1)\, t < \tau \leq it]$

By conditioning on when critical failure occurs and recognizing that the system is reset (repaired) at n_M t, one can write

$$MTBCF = \sum_{i=1}^{n_M} \tau_i [MCSP((i-1)t) - MCSP(it)] + MCSP(n_M t)(n_M t + MTBCF) \tag{39}$$

Using the approximation $\tau_i \doteq (i-\frac{1}{2})t$ gives

$$MTBCF = t[\tfrac{1}{2} + \sum_{i=1}^{n_{M-1}} MCSP(it) + \tfrac{1}{2}\, MCSP(n_M t)]/[1 - MCSP(n_M t)] \tag{40}$$

To avoid long processing times, n_M is limited to 20 in the MIREM software. When T_M exceeds 20 missions, the interval t in Equation 40 is adjusted.

To compute MTBMA, repair is viewed as a renewal process. Three events can occur at renewal:

CF: a critial failure

S: scheduled maintenance when there are no faults to repair

R: scheduled maintenance when there are repairs

Only the events CF and R are counted as maintenance actions. Let τ_{CF} denote the mean operating time since scheduled maintenance at which CF occurs. The total rate at which the events CF and R occur is

$$MTBMA^{-1} = MTBCF^{-1} + \frac{\Pr\{R \text{ at renewal}\}}{\tau_{CF} \Pr\{CF \text{ at renewal}\} + T_M(1-\Pr\{CF \text{ at renewal}\})}$$

$$= MTBCF^{-1} + \frac{MCSP(T_M) - e^{-T_M/MTBF}}{\tau_{CF}[1-MCSP(T_M)] + T_M\, MCSP(T_M)} \qquad (41)$$

The value of τ_{CF} can be determined from the rate of CF events:

$$MTBCF^{-1} = \frac{1 - MCSP(T_M)}{\tau_{CF}[1 - MCSP(T_M)] + T_M\, MCSP(T_M)}$$

$$\tau_{CF} = MTBCF - T_M\, MCSP(T_M)/[1 - MCSP(T_M)] \qquad (42)$$

Combining Equations 41 and 42 gives

$$MTBMA = \frac{MTBCF\ [1-MCSP(T_M)]}{1 - e^{-T_M/MTBF}} \qquad (43)$$

A.8.5 Repair at Degraded Level

For repair at degraded level, let

m_i = repair level for pool i

The system is repaired when $C_i(t) < m_i$ for some pool i. The reliability measures MCSP and MTBCF will be derived only for systems with series chains and no groups. First, note that $r_i \leq m_i \leq c_{max,i}$; i.e., repair must occur when a critical failure occurs, if not before. Let

$$p_i(x_i|m) = Pr\{C_i(t) = x_i \mid \underline{C}(t) \geq m\}$$

$$p_i(\underline{x}|\underline{m}) = Pr\{\underline{C}(t) = \underline{x} \mid \underline{C}(t) \geq \underline{m}\}, \text{ for a system with a steady-state distribution of } \underline{C}(0).$$

$$PF_i(x_i) = Pr\{C_i(t) < r_i \mid C_i(0) = x_i\}, \text{ calculated using Section A.3.}$$

Assuming the pool failures probabilities are relatively small, their higher-order terms will be dropped:

$$\overline{MCSP} = \sum_{\underline{r} \leq \underline{x} \leq \underline{m}} p(\underline{x}|\underline{m}) \prod_{\substack{\text{pools} \\ i}} [1-PF(x_i)]$$

$$\doteq \sum_{\underline{r} \leq \underline{x} \leq \underline{m}} p(\underline{x}|\underline{m}) \left[1- \sum_{\substack{\text{pools} \\ i}} PF(x_i)\right]$$

$$= 1 - \sum_{\substack{\text{pools} \\ i}} \sum_{\underline{r} \geq \underline{x} \geq \underline{m}} p(\underline{x}|\underline{m})\ PF_i(x_i)$$

$$= 1 - \sum_{\substack{\text{pools} \\ i}} \sum_{x_i = m_i} p_i(x_i|m)\ PF_i(x_i) \qquad (44)$$

To evaluate $p_i(x_i|\underline{m})$, the distribution of C_i under the repair policy $\underline{m}$, drop the pool subscript and let

$$p_j = p_i(j|\underline{m})$$

$$\underline{p} = [p_j],\ j = c_{max,i},\ldots,\ m_i \text{ (decreasing order)}$$

$$MTBMA_i = \text{MTBMA for this repair policy neglecting pool } i \text{ failures}$$

$$\lambda_R = 1/MTBMA_i = \text{repair rate without pool } i$$

$$\lambda = \text{pool } i \text{ branch failure rate}$$

Then C(t) can be represented by a continuous-time Markov renewal process if renewals (repairs) due to failures in other pools are approximated by a constant rate λ_R. The transition intensity matrix on the states $(c_{max},\ldots,\ m)$ for active redundant pools is

$$Q = \begin{bmatrix} -c_{max}\lambda & c_{max}\lambda & & & \\ \lambda_R & -\lambda_R-(c_{max}-1)\lambda & (c_{max}-1)\lambda & & \\ \cdot & \cdot & \cdot & & \\ \cdot & \cdot & \cdot & \cdot & \\ \cdot & \cdot & \cdot & \cdot & \\ & & -\lambda_R-(m+1)\lambda & (m+1)\lambda & \\ \lambda_R+m\lambda & & & -\lambda_R-m\lambda & \end{bmatrix} \qquad (45)$$

The steady-state distribution of C(t) is $\underline{p}$; it satisfies

$$Q^t\ \underline{p} = 0 \qquad (46)$$

which can be written

$$p_j = p_{j-1}\ (j-1 + \lambda_R/\lambda)/j,\ j = m+1,\ldots,\ c_{max}-1$$

$$p_{c_{max}} = [(p_{c_{max}-1} + \ldots + p_{m+1})(\lambda_R/\lambda) + p_m(m + \lambda_R/\lambda)]/n \qquad (47)$$

For standby redundant pools,

$$Q = \begin{bmatrix} -r\lambda & r\lambda & & & \\ \lambda_R & -\lambda_R - r\lambda & r\lambda & & \\ & \ddots & \ddots & \ddots & \\ & & & & r\lambda \\ \lambda_R + r\lambda & & & & -\lambda_R - r\lambda \end{bmatrix} \qquad (48)$$

and

$$p_j = p_m[1 + \lambda_R/(r\,\lambda)]^{j-m},\ j = m+1, \ldots, c_{max}-1$$

$$p_{c_{max}} = [(p_{c_{max}-1} + \ldots + p_{m+1})\lambda_R/\lambda + p_m(r + \lambda_R/\lambda)]/r \qquad (49)$$

Equation 47 or Equation 49 can be used to recursively compute p, given the normalization condition $p_m + \ldots + p_{c_{max}} = 1$.

MTBMA can be computed for systems with series or parallel chains. For consistency with the single utilization rates entered for type S pool pairs, MIREM uses a single repair level m_i for a type S pool pair. It is allocated to the individual pools as $\min\{m_i/2, c_{max,i}\}$. The MTBCF calculation of Section A.6 is then performed, treating all chains as series chains and using pool requirements of $\underline{m}$. This MTBCF is MTBMA for the repair policy $\underline{m}$.

A.9 Other Outputs

Several other quantities can be computed by MIREM. The MTBFF for function j is computed in the same manner as MTBCF using j as the only critical function; i.e., $F_1 = \{j\}$ and m=1.

The contribution of each LRM/LRU to reliability and maintenance actions can also be computed. Define the quantities

$$p_h(t) = \Pr\{\text{mission success at } t \mid \text{no failures in LRU h}\}$$

$q_h(t)$ = Pr{mission success at t discouting LRU h failures | mission failure by t}

MCSP(T;t) = Pr{mission success at t | mission success at t - T}

Λ_h = total failure rate in LRU h

$\bar{F}_h(t)$ = Pr{no faults in LRU h at time t}

$r_h(T;t)$ = Pr{fault in LRU h at time t | mission failure between t - T and t}

Then, neglecting standby redundancy,

$$\bar{F}_h(t) = \exp(-\Lambda_h t) \tag{50}$$

To compute $p_h(t)$, the failure rates λ in Equation 4 are set to zero for pools in LRU h; MCSP calculated under these conditions is equal to $p_h(t)$. Then

$$q_h(t) = 1 - \frac{1 - p_h(t)}{1 - MCSP} \tag{51}$$

The MCSP over a mission of length T for a system that has operated t - T hours without repair is

$$MCSP(T;t) = MCSP(t)/MCSP(t - T) \tag{52}$$

The last quantity of interest is $r_h(T;t)$, the probability of removing LRU h upon repair after an operating time of t:

$$r_h(T;t) = 1 - \frac{[p_h(t - T) - p_h(t)]\bar{F}_h(t)}{MCSP(t - T) - MCSP(t)} \tag{53}$$

APPENDIX B: SAMPLE SESSION LISTING

```
Welcome to the

                    M     M   III   RRR     EEEEE   M     M
                    M M M M    I    R  R    E       M M M M
                    M  M  M    I    RRR     EEE     M  M  M
                    M     M    I    R  R    E       M     M
                    M     M   III   R   R   EEEEE   M     M

       DDD       A     TTTTT     A         EEEEE N   N TTTTT RRR   Y   Y
       D  D     A A      T      A A        E     NN  N   T   R  R   Y Y
       D   D   AAAAA     T     AAAAA       EEE   N N N   T   RRR     Y
       D  D   A     A    T    A     A      E     N  NN   T   R  R    Y
       DDD   A       A   T   A       A     EEEEE N   N   T   R   R   Y

program.  (Version 03.86)

To obtain an explanation of keywords enter: <H>ELP
Or enter a command:<C>ONTINUE, <Q>UIT

c
                              DIALOGUE SELECTION MENU

Do you wish to:
   1.  Create the Architecture file from scratch.
   2.  Update an existing Architecture file.
   3.  Create the Scenario file from scratch. (Requires an Architecture file)
   4.  Update an existing Scenario file.

Please enter the index corresponding to your selection.
Or enter a command:<H>ELP, <E>XPLAIN, <Q>UIT

2
                              ARCHITECTURE FILE NAME

                    Index Description        Current Value
                    ----- -----------------  -------------
                      1.  Architecture File

Please enter the name of the Architecture File to be read.  1=filename
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

1=archin
                              ARCHITECTURE FILE NAME

                    Index Description        Current Value
                    ----- -----------------  -------------
                     *1.  Architecture File  ARCHIN

Please enter the name of the Architecture File to be read.  1=filename
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

c
```

ARCHITECTURE FILE MENU

Which type of data do you wish to work on?

Index	Selection	Dependencies	Number
1.	Functions	None	5
2.	LRMs/LRUs	None	3
3.	Resources	None	1
4.	Chains	Functions	2
5.	Pools	Functions, LRMs/LRUs, Resources, and Chains	1
6.	Save	Not Applicable	-

Please enter the index corresponding to your selection.
Or enter a command:<H>ELP, <E>XPLAIN, <B>ACK, <Q>UIT

1

FUNCTION LIST PAGE 1 OF 2

Index	Function	Index	Function	Index	Function
1.	FUNC1	2.	FCN2	3.	FCN3
4.		5.	FCN5	6.	
7.		8.		9.	
10.		11.		12.	
13.		14.		15.	
16.		17.		18.	
19.		20.		21.	
22.		23.		24.	
25.		26.		27.	
28.		29.		30.	
31.		32.		33.	
34.		35.		36.	
37.		38.		39.	

Please enter the command: index=value, index=value, ...
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

4=fcn4

FUNCTION LIST PAGE 1 OF 2

Index	Function	Index	Function	Index	Function
1.	FUNC1	2.	FCN2	3.	FCN3
* 4.	FCN4	5.	FCN5	6.	
7.		8.		9.	
10.		11.		12.	
13.		14.		15.	
16.		17.		18.	
19.		20.		21.	
22.		23.		24.	
25.		26.		27.	
28.		29.		30.	
31.		32.		33.	
34.		35.		36.	
37.		38.		39.	

Please enter the command: index=value, index=value, ...
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

c

ARCHITECTURE FILE MENU

Which type of data do you wish to work on?

Index	Selection	Dependencies	Number
1.	Functions	None	5
2.	LRMs/LRUs	None	3
3.	Resources	None	1
4.	Chains	Functions	2
5.	Pools	Functions, LRMs/LRUs, Resources, and Chains	1
6.	Save	Not Applicable	-

Please enter the index corresponding to your selection.
Or enter a command:<H>ELP, <E>XPLAIN, <B>ACK, <Q>UIT

2

LRM/LRU LIST PAGE 1 OF 6

Index	LRM/LRU Name	Index	LRM/LRU Name	Index	LRM/LRU Name
1.	LRM1	2.	LRU1	3.	LRU2
4.		5.		6.	
7.		8.		9.	
10.		11.		12.	
13.		14.		15.	
16.		17.		18.	
19.		20.		21.	
22.		23.		24.	
25.		26.		27.	
28.		29.		30.	
31.		32.		33.	
34.		35.		36.	
37.		38.		39.	

Please enter the command: index=value, index=value, ...
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

4=lrm2 5=lru3

LRM/LRU LIST PAGE 1 OF 6

Index	LRM/LRU Name	Index	LRM/LRU Name	Index	LRM/LRU Name
1.	LRM1	2.	LRU1	3.	LRU2
* 4.	LRM2	* 5.	LRU3	6.	
7.		8.		9.	
10.		11.		12.	
13.		14.		15.	
16.		17.		18.	
19.		20.		21.	
22.		23.		24.	
25.		26.		27.	
28.		29.		30.	
31.		32.		33.	
34.		35.		36.	
37.		38.		39.	

Please enter the command: index=value, index=value, ...
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

c

ARCHITECTURE FILE MENU

Which type of data do you wish to work on?

Index	Selection	Dependencies	Number
1.	Functions	None	5
2.	LRMs/LRUs	None	5
3.	Resources	None	1
4.	Chains	Functions	2
5.	Pools	Functions, LRMs/LRUs, Resources, and Chains	1
6.	Save	Not Applicable	-

Please enter the index corresponding to your selection.
Or enter a command:<H>ELP, <E>XPLAIN, <B>ACK, <Q>UIT

3

RESOURCE LIST

Index	Res. No.	Qty	Failure Rate	R/I	MTTR	Resource Name	Change/ Delete
1.	1	1	5	R	1.00	L-BAND RECEIVER	

NOTE: R = Resource, I = Interconnection
MTTR = Mean Time To Repair (hours)

Please enter the command: index=value, index=value, ...
(values are 'c' for Change and 'd' for Delete)
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <A>DD, <B>ACK, <Q>UIT

1=c

RESOURCE LIST

Index	Res. No.	Qty	Failure Rate	R/I	MTTR	Resource Name	Change/ Delete
*1.	1	1	5	R	1.00	L-BAND RECEIVER	C

NOTE: R = Resource, I = Interconnection
MTTR = Mean Time To Repair (hours)

Please enter the command: index=value, index=value, ...
(values are 'c' for Change and 'd' for Delete)
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <A>DD, <B>ACK, <Q>UIT

c

RESOURCE DATA ENTRY

Index	Parameter	Current Value
1.	Resource Number	1
2.	Quantity	1
3.	Failure rate (x E-6 Hours)	5
4.	Resource or Interconnection (R or I)	R
5.	Mean time to repair (hrs)	1.00
6.	Resource Name	L-BAND RECEIVER

Please enter the command: index=value, index=value, ...
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

2=2 3=10

RESOURCE DATA ENTRY

Index	Parameter	Current Value
1.	Resource Number	1
*2.	Quantity	2
*3.	Failure rate (x E-6 Hours)	10
4.	Resource or Interconnection (R or I)	R
5.	Mean time to repair (hrs)	1.00
6.	Resource Name	L-BAND RECEIVER

Please enter the command: index=value, index=value, ...
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

c

RESOURCE LIST

Index	Res. No.	Qty	Failure Rate	R/I	MTTR	Resource Name	Change/ Delete
1.	1	2	10	R	1.00	L-BAND RECEIVER	

NOTE: R = Resource, I = Interconnection
MTTR = Mean Time To Repair (hours)

Please enter the command: index=value, index=value, ...
(values are 'c' for Change and 'd' for Delete)
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <A>DD, <B>ACK, <Q>UIT

c

ARCHITECTURE FILE MENU

Which type of data do you wish to work on?

Index	Selection	Dependencies	Number
1.	Functions	None	5
2.	LRMs/LRUs	None	5
3.	Resources	None	1
4.	Chains	Functions	2
5.	Pools	Functions, LRMs/LRUs, Resources, and Chains	1
6.	Save	Not Applicable	-

Please enter the index corresponding to your selection.
Or enter a command:<H>ELP, <E>XPLAIN, <B>ACK, <Q>UIT

4

CHAIN LIST

Index	Chain Number	Parallel Chain Number	Chain Pair Name	Change/ Delete
1.	1	None	SERIES CHAIN	
2.	2	3	PARALLEL CHAIN	

Please enter the command: index=value, index=value, ...
(values are 'c' for Change and 'd' for Delete)
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <A>DD, <B>ACK, <Q>UIT

1=c 2=c c

CHAIN DATA ENTRY

Index	Parameter	Current Value
1.	Chain Number	1
2.	Parallel Chain Number	
3.	Chain Name	SERIES CHAIN

Please enter the command: index=value, index=value, ...
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

c

FUNCTIONS IN CHAIN NUMBER 1

Index	Function
*1.	FUNC1
2.	FCN2
*3.	FCN3
4.	FCN4
*5.	FCN5

NOTE: The asterisk ('*') indicates functions selected.

Please enter the command: index to select or -index to delete.
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

2

FUNCTIONS IN CHAIN NUMBER 1

Index	Function
*1.	FUNC1
*2.	FCN2
*3.	FCN3
4.	FCN4
*5.	FCN5

NOTE: The asterisk ('*') indicates functions selected.

Please enter the command: index to select or -index to delete.
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

c

CHAIN DATA ENTRY

Index	Parameter	Current Value
1.	Chain Number	2
2.	Parallel Chain Number	3
3.	Chain Name	PARALLEL CHAIN

Please enter the command: index=value, index=value, ...
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

c

FUNCTIONS IN CHAIN NUMBER 2

Index	Function
*1.	FUNC1
*2.	FCN2
*3.	FCN3
4.	FCN4
5.	FCN5

NOTE: The asterisk ('*') indicates functions selected.

Please enter the command: index to select or -index to delete.
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

c

FUNCTIONS IN CHAIN NUMBER 3

Index	Function
*1.	FUNC1
*2.	FCN2
*3.	FCN3
4.	FCN4
5.	FCN5

NOTE: The asterisk ('*') indicates functions selected.

Please enter the command: index to select or -index to delete.
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

c

CHAIN LIST

Index	Chain Number	Parallel Chain Number	Chain Pair Name	Change/ Delete
1.	1	None	SERIES CHAIN	
2.	2	3	PARALLEL CHAIN	

Please enter the command: index=value, index=value, ...
(values are 'c' for Change and 'd' for Delete)
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <A>DD, <B>ACK, <Q>UIT

c

ARCHITECTURE FILE MENU

Which type of data do you wish to work on?

Index	Selection	Dependencies	Number
1.	Functions	None	5
2.	LRMs/LRUs	None	5
3.	Resources	None	1
4.	Chains	Functions	2
5.	Pools	Functions, LRMs/LRUs, Resources, and Chains	1
6.	Save	Not Applicable	-

Please enter the index corresponding to your selection.
Or enter a command:<H>ELP, <E>XPLAIN, <B>ACK, <Q>UIT

5

POOL LIST

Index	Pool No.	Chain No.	Lrm/Lru Name	Pool Type	No. of Branches	Resource Numbers	A/S	C/R/D
1.	1	2	LRU1	Noncontending	1	1	A	

Please enter the command: index=value, index=value, ...
(values are 'c' for Change, 'r' for Repeat, and 'd' for Delete)
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <A>DD, <B>ACK, <Q>UIT

1=r

POOL LIST

Index	Pool No.	Chain No.	Lrm/Lru Name	Pool Type	No. of Branches	Resource Numbers	A/S	C/R/D
*1.	1	2	LRU1	Noncontending	1	1	A	R

Please enter the command: index=value, index=value, ...
(values are 'c' for Change, 'r' for Repeat, and 'd' for Delete)
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <A>DD, <B>ACK, <Q>UIT

c

```
                                POOL LIST

 Index  Pool No.  Chain No.  Lrm/Lru Name      Pool Type       No. of Branches  Resource Numbers  A/S  C/R/D
 -----  --------  ---------  ----------------  --------------  ---------------  ----------------  ---  -----
   1.       1          2    LRU1              Noncontending          1          1                A
   2.       1          3    LRU1              Noncontending          1          1                A

Please enter the command: index=value, index=value, ...
     (values are 'c' for Change, 'r' for Repeat, and 'd' for Delete)
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <A>DD, <B>ACK, <Q>UIT

2=d c
                                POOL LIST

 Index  Pool No.  Chain No.  Lrm/Lru Name      Pool Type       No. of Branches  Resource Numbers  A/S  C/R/D
 -----  --------  ---------  ----------------  --------------  ---------------  ----------------  ---  -----
   1.       1          2    LRU1              Noncontending          1          1                A

Please enter the command: index=value, index=value, ...
     (values are 'c' for Change, 'r' for Repeat, and 'd' for Delete)
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <A>DD, <B>ACK, <Q>UIT

1=r
                                POOL LIST

 Index  Pool No.  Chain No.  Lrm/Lru Name      Pool Type       No. of Branches  Resource Numbers  A/S  C/R/D
 -----  --------  ---------  ----------------  --------------  ---------------  ----------------  ---  -----
  *1.       1          2    LRU1              Noncontending          1          1                A    R

Please enter the command: index=value, index=value, ...
     (values are 'c' for Change, 'r' for Repeat, and 'd' for Delete)
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <A>DD, <B>ACK, <Q>UIT

c
                            POOL DATA ENTRY

          Index  Parameter                          Current Value
          -----  ------------------------------     -------------
            1.   Pool Number                          1
            2.   Chain Number
            3.   LRM/LRU Name                        LRU1
            4.   Pool Type ('N', 'C', 'S', 'F')     N
            5.   Number of Branches                   1
            6.   Active/Standby ('A' or 'S')         A
            7.   Undetected Failure Rate             0.010
            8.   False Alarm Rate                    0.050
            9.   Min accpt level of repair          1

NOTE: (4). 'N' is Noncontending, 'C' is Contending, 'S' is Shared, and
           'F' is Chain-Fail.

Please enter the command: index=value, index=value, ...
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

2=3
```

POOL DATA ENTRY

Index	Parameter	Current Value
1.	Pool Number	1
*2.	Chain Number	3
3.	LRM/LRU Name	LRU1
4.	Pool Type ('N', 'C', 'S', 'F')	N
5.	Number of Branches	1
6.	Active/Standby ('A' or 'S')	A
7.	Undetected Failure Rate	0.010
8.	False Alarm Rate	0.050
9.	Min accpt level of repair	1

NOTE: (4). 'N' is Noncontending, 'C' is Contending, 'S' is Shared, and 'F' is Chain-Fail.

Please enter the command: index=value, index=value, ...
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

c

POOL RESOURCES FOR POOL NUMBER 1 PAGE 1 OF 2

Index	Resource Name	Resource Number
1.	L-BAND RECEIVER	1
2.		
3.		
4.		
5.		
6.		
7.		
8.		
9.		
10.		
11.		
12.		
13.		

Please enter the command: index=value, index=value, ...
Or enter a command:<H>ELP, <E>XPLAIN, <P>AGEn, <C>ONTINUE, <B>ACK, <Q>UIT

c

UTILIZATION RATES FOR POOL NUMBER 1

Index	Function	Rate
1.	FUNC1	1.00
2.	FCN2	0.00
3.	FCN3	0.00
4.	FCN4	0.00
5.	FCN5	0.00

Please enter the command: index=value, index=value, ...
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

1=0 2=1

UTILIZATION RATES FOR POOL NUMBER 1

Index	Function	Rate
*1.	FUNC1	0.00
*2.	FCN2	1.00
3.	FCN3	0.00
4.	FCN4	0.00
5.	FCN5	0.00

```
Please enter the command: index=value, index=value, ...
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

c
```

POOL LIST

Index	Pool No.	Chain No.	Lrm/Lru Name	Pool Type	No. of Branches	Resource Numbers	A/S	C/R/D
1.	1	2	LRU1	Noncontending	1	1	A	
2.	1	3	LRU1	Noncontending	1	1	A	

```
Please enter the command: index=value, index=value, ...
     (values are 'c' for Change, 'r' for Repeat, and 'd' for Delete)
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <A>DD, <B>ACK, <Q>UIT

c
```

ARCHITECTURE FILE MENU

Which type of data do you wish to work on?

Index	Selection	Dependencies	Number
1.	Functions	None	5
2.	LRMs/LRUs	None	5
3.	Resources	None	1
4.	Chains	Functions	2
5.	Pools	Functions, LRMs/LRUs, Resources, and Chains	2
6.	Save	Not Applicable	-

```
Please enter the index corresponding to your selection.
Or enter a command:<H>ELP, <E>XPLAIN, <B>ACK, <Q>UIT

6
```

ARCHITECTURE FILE NAME

Index	Description	Current Value
1.	Architecture File	

```
Please enter the name of the Architecture File to be created.  1=filename
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

1=archout c
```

```
                    DIALOGUE SELECTION MENU

Do you wish to:
   1.  Create the Architecture file from scratch.
   2.  Update an existing Architecture file.
   3.  Create the Scenario file from scratch. (Requires an Architecture file)
   4.  Update an existing Scenario file.

Please enter the index corresponding to your selection.
Or enter a command:<H>ELP, <E>XPLAIN, <Q>UIT

4
                        SCENARIO FILE NAME

                Index Description        Current Value
                ----- -----------------  -------------
                  1.  Scenario File

Please enter the name of the Scenario File to be read.  1=filename
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

1=scenin
                        SCENARIO FILE NAME

                Index Description        Current Value
                ----- -----------------  -------------
                 *1.  Scenario File      SCENIN

Please enter the name of the Scenario File to be read.  1=filename
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

c
                      ARCHITECTURE FILE NAME

                Index Description        Current Value
                ----- -----------------  -------------
                  1.  Architecture File  ARCHIN

Please enter the name of the Architecture File to be read.  1=filename
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

c
                          RUN IDENTIFIER

A Run Identifier is placed at the top of each printout page as a means of
distinguishing results from various runs.  It contains up to 72 characters.

Index Run Identifier
----- ------------------------------------------------------------------
  1.  SCENARIO FILE EDITING TEST1

Please enter the Run Identifier.  1='run identifier'
Or enter a command:<H>ELP, <C>ONTINUE, <B>ACK, <Q>UIT

1='scenario file editing test2'
```

```
                                RUN IDENTIFIER

A Run Identifier is placed at the top of each printout page as a means of
distinguishing results from various runs.  It contains up to 72 characters.

Index Run Identifier
-----  ---------------------------------------------------------------------
 *1.  SCENARIO FILE EDITING TEST2

Please enter the Run Identifier.  1='run identifier'
Or enter a command:<H>ELP, <C>ONTINUE, <B>ACK, <Q>UIT

c
                            COMPUTATION SELECTION MENU

Which do you wish to compute? :
             *1.  MCSP And Pool/Chain Budget; Immediate Repair MTBCF
             *2.  Phase-By-Phase MCSP
             *3.  MTBCF
             *4.  MTBFF
             *5.  LRM/LRU Budget
             *6.  Repair
             *7.  BIT Effectiveness
             *8.  Full BIT Effectiveness

NOTE: MCSP  - Mission Completion Success Probability
      MTBCF - Mean Time Between Critical Failures
      MTBFF - Mean Time Between Function Failures

NOTE: The asterisk ('*') indicates functions selected.

Please enter the command: index to select or -index to delete.
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

-2 -5
                            COMPUTATION SELECTION MENU

Which do you wish to compute? :
             *1.  MCSP And Pool/Chain Budget; Immediate Repair MTBCF
              2.  Phase-By-Phase MCSP
             *3.  MTBCF
             *4.  MTBFF
              5.  LRM/LRU Budget
             *6.  Repair
             *7.  BIT Effectiveness
             *8.  Full BIT Effectiveness

NOTE: MCSP  - Mission Completion Success Probability
      MTBCF - Mean Time Between Critical Failures
      MTBFF - Mean Time Between Function Failures

NOTE: The asterisk ('*') indicates functions selected.

Please enter the command: index to select or -index to delete.
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

c
```

PLOT SELECTION MENU

Which do you wish to plot? :
- *1. MCSP And Pool/Chain Budget; Immediate Repair MTBCF
- 2. Phase-By-Phase MCSP
- 3. MTBCF
- 4. MTBFF
- 5. LRM/LRU Budget
- 6. Repair

NOTE: MCSP - Mission Completion Success Probability
MTBCF - Mean Time Between Critical Failures
MTBFF - Mean Time Between Function Failures

NOTE: The asterisk ('*') indicates functions selected.

Please enter the command: index to select or -index to delete.
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

3 6

PLOT SELECTION MENU

Which do you wish to plot? :
- *1. MCSP And Pool/Chain Budget; Immediate Repair MTBCF
- 2. Phase-By-Phase MCSP
- *3. MTBCF
- 4. MTBFF
- 5. LRM/LRU Budget
- *6. Repair

NOTE: MCSP - Mission Completion Success Probability
MTBCF - Mean Time Between Critical Failures
MTBFF - Mean Time Between Function Failures

NOTE: The asterisk ('*') indicates functions selected.

Please enter the command: index to select or -index to delete.
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

c

TOTAL OPERATING TIME DATA ENTRY

Index	Function	Index	Function	Index	Function
1.	1.00	2.	1.50	3.	2.50
4.	4.00	5.		6.	
7.					

Please enter the command: index=value, index=value, ...
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

3=2.5

TOTAL OPERATING TIME DATA ENTRY

Index	Function	Index	Function	Index	Function
1.	1.00	2.	1.50	*3.	2.50
4.	4.00	5.		6.	
7.					

Please enter the command: index=value, index=value, ...
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

c

BASIC SCENARIO FILE PARAMETERS

Index	Parameter	Allowed Values	Current Value
1.	Processing Option	Quick, Full	FULL
2.	Print Architecture File Report?	Yes, No	YES
3.	Print Intermediate Results?	Yes, No	NO
4.	Functions Required Simultaneously?	Yes, No	YES
5.	Failure Rate Scale Factor	Positive Real Number	1.00
6.	Scheduled Maintenance (hours)	Positive Real Number	2.00
7.	Repair Sequence	Series or Parallel	P

Please enter the command: index=value, index=value, ...
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

1=quick 3=yes 4=no 6=2.5

BASIC SCENARIO FILE PARAMETERS

Index	Parameter	Allowed Values	Current Value
*1.	Processing Option	Quick, Full	QUICK
2.	Print Architecture File Report?	Yes, No	YES
*3.	Print Intermediate Results?	Yes, No	YES
*4.	Functions Required Simultaneously?	Yes, No	NO
5.	Failure Rate Scale Factor	Positive Real Number	1.00
*6.	Scheduled Maintenance (hours)	Positive Real Number	2.50
7.	Repair Sequence	Series or Parallel	P

Please enter the command: index=value, index=value, ...
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

c

MISSION PHASE LIST

Index	Phase Number	Length (Hours)	Phase Name	Critical Functions	Change/ Delete
1.	1	0.33	FRONT END	1	
2.	2	0.67	BACK END	2	

Please enter the command: index=value, index=value, ...
(values are 'c' for Change and 'd' for Delete)
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <A>DD, <B>ACK, <Q>UIT

1=c

MISSION PHASE LIST

Index	Phase Number	Length (Hours)	Phase Name	Critical Functions	Change/ Delete
*1.	1	0.33	FRONT END	1	C
2.	2	0.67	BACK END	2	

```
Please enter the command: index=value, index=value, ...
     (values are 'c' for Change and 'd' for Delete)
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <A>DD, <B>ACK, <Q>UIT

c
```

MISSION PHASE DATA ENTRY

Index	Parameter	Current Value
1.	Mission Phase Number	1
2.	Mission Phase Length (Hours)	0.33
3.	Mission Phase Name	FRONT END

```
Please enter the command: index=value, index=value, ...
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

2=.2
```

MISSION PHASE DATA ENTRY

Index	Parameter	Current Value
1.	Mission Phase Number	1
*2.	Mission Phase Length (Hours)	0.20
3.	Mission Phase Name	FRONT END

```
Please enter the command: index=value, index=value, ...
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

c
```

CRITICAL FUNCTIONS FOR MISSION PHASE 1

Index	Function
*1.	FUNC1
2.	FCN2
3.	FCN3
4.	
5.	FCN5

```
NOTE: The asterisk ('*') indicates functions selected.

Please enter the command: index to select or -index to delete.
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

3
```

CRITICAL FUNCTIONS FOR MISSION PHASE 1

Index	Function
*1.	FUNC1
2.	FCN2
*3.	FCN3
4.	
5.	FCN5

NOTE: The asterisk ('*') indicates functions selected.

Please enter the command: index to select or -index to delete.
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

c

MISSION PHASE LIST

Index	Phase Number	Length (Hours)	Phase Name	Critical Functions	Change/ Delete
1.	1	0.20	FRONT END	2	
2.	2	0.67	BACK END	2	

Please enter the command: index=value, index=value, ...
(values are 'c' for Change and 'd' for Delete)
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <A>DD, <B>ACK, <Q>UIT

c

SCENARIO FILE NAME

Index	Description	Current Value
1.	Scenario File	

Please enter the name of the Scenario File to be created. 1=filename
Or enter a command:<H>ELP, <E>XPLAIN, <C>ONTINUE, <B>ACK, <Q>UIT

1=scenout c

DIALOGUE SELECTION MENU

Do you wish to:
1. Create the Architecture file from scratch.
2. Update an existing Architecture file.
3. Create the Scenario file from scratch. (Requires an Architecture file)
4. Update an existing Scenario file.

Please enter the index corresponding to your selection.
Or enter a command:<H>ELP, <E>XPLAIN, <Q>UIT

q

TERMINATION SCREEN

You have requested to end the MIREM Data Entry program now in progress. To stop the current processing just enter q or quit. NOTE, however, that this will cause any changes made since the last file creation to be lost.

If you have not stored the most recent changes made, but wish to do so, enter c or continue and you can continue processing from the point that this screen was entered. After you have stored the file being edited, you may then terminate the program safely by entering q or quit again.

Please enter the command:<C>ONTINUE, <Q>UIT

q

MESSAGE SUMMARY: MESSAGE NUMBER - COUNT

219 2

DASD 123 DETACHED

APPENDIX C: DATA ENTRY FORMS

A-15380a

POOL DATA SHEET

Date: ______________ Page _____ of _____

System: ______________ Architecture File Name: ______________ Analyst: ______________

POOL	POOL NO.	CHAIN NO:	LRU INDEX:	POOL TYPE:	NO. BRANCHES:	ACTIVE/STANDBY: UNDETECTED FAILURES:	FALSE ALARMS:	MIN REPAIR LEVEL:
POOL RESOURCE								
POOL UTIL.								

POOL	POOL NO.	CHAIN NO:	LRU INDEX:	POOL TYPE:	NO. BRANCHES:	ACTIVE/STANDBY: UNDETECTED FAILURES:	FALSE ALARMS:	MIN REPAIR LEVEL:
POOL RESOURCE								
POOL UTIL.								

POOL	POOL NO.	CHAIN NO:	LRU INDEX:	POOL TYPE:	NO. BRANCHES:	ACTIVE/STANDBY: UNDETECTED FAILURES:	FALSE ALARMS:	MIN REPAIR LEVEL:
POOL RESOURCE								
POOL UTIL.								

POOL	POOL NO.	CHAIN NO:	LRU INDEX:	POOL TYPE:	NO. BRANCHES:	ACTIVE/STANDBY: UNDETECTED FAILURES:	FALSE ALARMS:	MIN REPAIR LEVEL:
POOL RESOURCE								
POOL UTIL.								

POOL	POOL NO.	CHAIN NO:	LRU INDEX:	POOL TYPE:	NO. BRANCHES:	ACTIVE/STANDBY: UNDETECTED FAILURES:	FALSE ALARMS:	MIN REPAIR LEVEL:
POOL RESOURCE								
POOL UTIL.								

POOL	POOL NO.	CHAIN NO:	LRU INDEX:	POOL TYPE:	NO. BRANCHES:	ACTIVE/STANDBY: UNDETECTED FAILURES:	FALSE ALARMS:	MIN REPAIR LEVEL:
POOL RESOURCE								
POOL UTIL.								

RESOURCE DATA SHEET

A 15263

Date:_____ Page____of____

System:_____

Architecture File Name:_____ Analyst:_____

RESOURCE NO.	RESOURCE NAME	R/I	MTTR	QUANTITY	FAILURE RATE

APPENDIX D: GLOSSARY

BIT	Built-In Test.
branch	A simple parallel path in a pool.
chain	A set of pools arranged in series. Chains are organized into a series/parallel structure.
chain-fail pool	Also type F pool. A pool that, upon failure prevents all resources in a chain from being used, including type S resources. Chain-fail pools are utilized in a noncontending fashion.
chain pair	Two parallel chains. Also used to refer to a series chain, so that all chains can be indexed by chain pair.
contending pool	Also type C pool. A pool in which separate resources, or fractions thereof, must be allocated to each function using the pool.
critical failure	A failure that causes loss of a critical function, and hence, loss of mission capability.
critical function	A function that is required during a mission phase. Criticality can be defined in terms of mission success, survival, etc.
failure resiliency	MTBCF/MTBF, a measure of fault tolerance.
function	A distinct operational capability of the system. Used to describe system requirements.
group	A k-of-n structure containing other groups or pools.
GPS	Global positioning system.
ICNIA	Integrated communication, navigation and identification avionics.
LRM	Line replaceable module.
LRU	Line replaceable unit.

MCSP	Mission completion success probability; the probability of completing a mission, or any specified time of operation, without a critical failure.
MTBCF	Mean time between critical failure.
MTBF	Mean time between failure (first failure).
MTBFF	Mean time between function failure, for an individual function.
MTBMA	Mean time between maintenance actions.
MTTR	Mean time to repair.
noncontending pool	Also type N pool. A pool in which the same resources can be utilized by any number of functions.
parallel chain	A chain in parallel with another chain; functions are allocated to one of the two chains.
phase	A time interval within a mission; different functions are required in each phase.
pool	A reliability structure consisting of one or more simple parallel branches.
pool pair	Two type C or S pools, one in each chain of a parallel chain pair, with the same utilization rates and the same pool number.
pool type	The manner in which a resource pool is utilized by functions: noncontending (N), contending (C), shared (S) or chain-fail (F).
primary chain	The first chain in a chain pair.
primary pool	The pool in a pool pair that is in the primary chain.
RBD	Reliability block diagram.
resource	A component or portion of the system that fails as a unit and has the same status (faulty or good) with respect to all functions that can use the resource.

SDU	Secure data unit.
secondary chain	The second chain in a parallel chain pair.
secondary pool	The pool in a pool pair that is in the secondary chain.
series chain	A chain that is not in parallel with another chain; it is in series with all other chain pairs.
shared pool	Also type S pool. A pool in a parallel chain that shares resources with its counterpart on the other chain in the parallel chain pair. Shared pools are utilized in a contending fashion.
SINCGARS	Single-channel ground and airborne radio subsystem.
SRU	Shop replacable unit.
Subgroup	A pool or group contained in a larger group.
UHF	Ultra-high frequency voice communication.
utilization rate	The number of branches in a pool a group or subgroups in required by a function; may be fractional for timeshared or multiplexed resources.

U.S. GOVERNMENT PRINTING OFFICE:1986-761-057/40033

END

2-87

DTIC